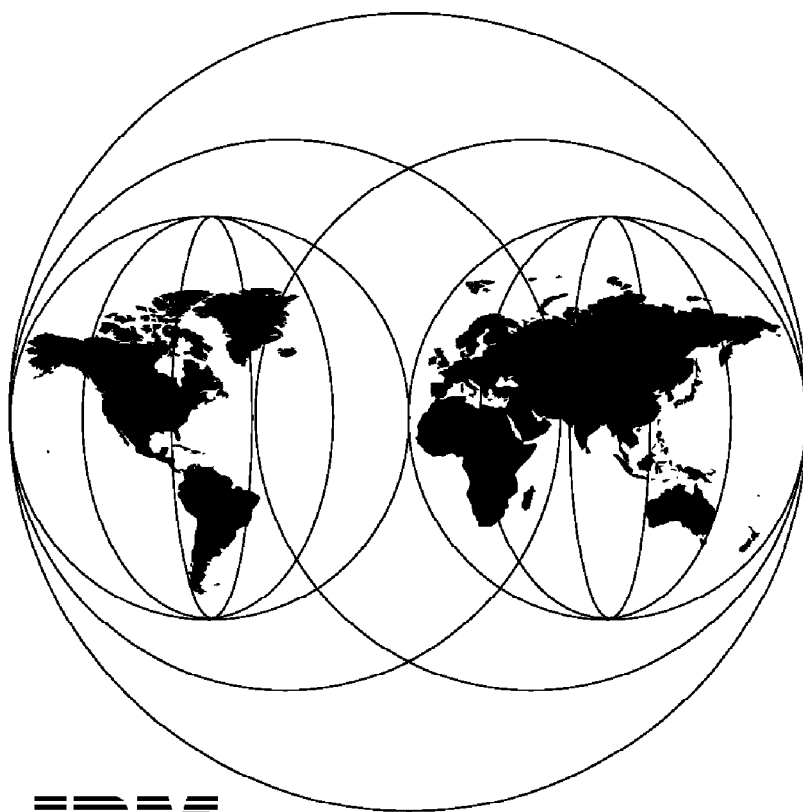


Host/Workstation Client/Server Implementation Using VisualAge COBOL on OS/2, AIX, and MVS

December 1996



IBM

**International Technical Support Organization
San Jose Center**



International Technical Support Organization

SG24-4733-00

**Host/Workstation Client/Server Implementation Using
VisualAge COBOL on OS/2, AIX, and MVS**

December 1996

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix A, "Special Notices" on page 235.

First Edition (December 1996)

This edition applies to:

- Version 1.2 Refresh, 83H9131, VisualAge for COBOL for OS/2 Standard Edition
- Version 1.0, 28H2176, COBOL Set for AIX
- Version 1.2, 5688-197, COBOL for MVS

for use with the Operating Systems:

- OS/2 WARP 3.0
- AIX 4.1
- MVS/ESA 5.2.

Comments may be addressed to:

IBM Corporation, International Technical Support Organization
Dept. QXXE Building 80-E2
650 Harry Road
San Jose, California 95120-6099

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1996. All rights reserved.

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Figures	v
Preface	xi
How This Redbook Is Organized	xi
The Team That Wrote This Redbook	xi
Comments Welcome	xii
Chapter 1. OS/2 Local Area Network with DB2	1
1.1 OS/2 Server Installation	2
1.1.1 Installing OS/2 LAN Server 4.0	3
1.1.2 Defining the Users of the Domain	8
1.1.3 Installing IBM DATABASE 2 for OS/2 Server	13
1.1.4 Configuring DB2 for OS/2 on the Server	18
1.2 OS/2 Client Installation	25
1.2.1 Installing DB2 for OS/2 - Single-User	31
1.2.2 Installation of Software Developer's Kit for OS/2	35
1.2.3 DB2 Client Setup	37
1.2.4 Installing IBM VisualAge for COBOL for OS/2	44
1.2.5 Executing a COBOL Application That Accesses a Remote Database	47
Chapter 2. OS/2 LAN with CICS and DB2	59
2.1 CICS for OS/2	60
2.1.1 CICS for OS/2 Installation	60
2.2 OS/2 CICS Client	69
2.2.1 CICS Client Installation	70
2.2.2 Using OS/2 CICS Client	74
2.3 Running CICS Client/Server Applications	77
2.3.1 Running the First CICS Client/Server Application	78
2.3.2 Running a CICS Client/Server Application with Database Access	84
2.3.3 How to Use the Transaction Assistant	102
2.3.4 Debugging IBM VisualAge for COBOL for OS/2 CICS Programs	106
Chapter 3. COBOL with DB2 for AIX and DB2 Client	109
3.1 DB2 for AIX	110
3.1.1 DB2 for AIX Server Installation	111
3.1.2 DB2 for AIX Server Configuration	116
3.2 Set up DB2 Client to Access DB2 for AIX	119
3.2.1 TCP/IP Installation	119
3.2.2 DB2 Client Configuration	125
3.3 Executing a COBOL Program That Accesses Database on AIX	129
Chapter 4. COBOL with CICS for AIX and CICS Client for OS/2	131
4.1 CICS for AIX Server Setup	132
4.1.1 Installing and Configuring IBM COBOL Set for AIX	132
4.1.2 Installing and Configuring CICS for AIX Server	133
4.1.3 Compiling and Linking the CICS Server Program	145
4.1.4 Defining CICS Resources for the CICS Server Program	147
4.2 CICS Client for OS/2 Environment Setup	150
4.2.1 Configuring CICS Client to Connect to CICS for AIX	150
4.2.2 Compiling and Linking the CICS Client Program	154
4.2.3 Executing the CICS Client/Server Programs	155

Chapter 5. Host Development Offload	157
5.1 Installing and Configuring Communications Manager/2 for OS/2 Warp	158
5.2 Download the Necessary Data to the Workstation	168
5.2.1 Source Code	169
5.2.2 Test Data	171
5.3 Prepare the Application on the Workstation	171
5.3.1 CICS 3270 Client Application	171
5.3.2 CICS-DB2 Server Application	184
5.4 Upload the Necessary Data to the MVS System	189
5.5 Prepare the Application on the MVS System	193
5.5.1 Adapt the Source Code for MVS	194
5.5.2 Build the MVS Application	198
5.5.3 CICS for MVS/ESA Definitions	204
5.5.4 Run the Transaction on the Host	208
 Chapter 6. OS/2 Client with MVS Server	 213
6.1 Overview	214
6.2 Configuring Communications Manager/2 for OS/2 Warp	215
6.2.1 APPC Set-up	216
6.2.2 Change Number of Sessions	225
6.3 Change the CICS Client Definitions	226
6.4 VTAM Definitions	227
6.4.1 NETID Definition	227
6.4.2 XID/PU/LU Definition	227
6.4.3 APPL Definition	228
6.4.4 LOGMODE Definition	228
6.4.5 VTAM Cross-Domain Environment	229
6.5 CICS for MVS/ESA Definitions	229
6.5.1 SIT Definitions	230
6.5.2 Connection/Session Definitions	230
6.5.3 Data Conversion Definitions	232
6.6 Run the Application	233
 Appendix A. Special Notices	 235
 Appendix B. Related Publications	 237
B.1 International Technical Support Organization Publications	237
B.2 Redbooks on CD-ROMs	237
B.3 Other Publications	237
 How To Get ITSO Redbooks	 239
How IBM Employees Can Get ITSO Redbooks	239
How Customers Can Get ITSO Redbooks	240
IBM Redbook Order Form	241
 Glossary	 243
 List of Abbreviations	 263
 Index	 265

Figures

1.	OS/2 LAN-Based Environment	1
2.	LAN Server Welcome Window	3
3.	CD-ROM Installation Window	4
4.	Server Installation/Configuration Window	4
5.	Easy or Tailored Installation/Configuration Window	5
6.	Installation Location Window	5
7.	Server Name Window	5
8.	Domain Name Window	6
9.	First Server—Domain Controller Window	6
10.	Network Adapter Confirmation Window	6
11.	Default User ID and Password Information Window	7
12.	Create a Startup Diskette For Your Workstation Window	7
13.	Create a Startup Diskette Window	8
14.	Installation Completion Message Window	8
15.	IBM LAN Services—Icon View	9
16.	LAN Server Logon Window	9
17.	Duplicate ID Message	9
18.	About Window	10
19.	LAN Server Administration - Icon View	10
20.	Domain - Icon View	10
21.	User Accounts - Icon View	11
22.	User Account - Create Window—Identity Page	11
23.	User Account - Create Window—Password Page	12
24.	User Account - Create Window—Privileges Page	12
25.	User Account - Delete Window	13
26.	IBM Database Server Window	14
27.	IBM Database Server for OS/2 - Installation Window	14
28.	Install Window (DB2 for OS/2 Server)	15
29.	Install - directories Window (DB2 Server)	15
30.	Product License Key Window	16
31.	Install - progress Window (DB2 Server)	16
32.	Instruction ! Window	16
33.	Using DB2 for the First Time Window (DB2 Server)	17
34.	Local Logon Window (DB2 Server)	17
35.	Target Drive Selection Window (DB2 Server)	18
36.	Sample Database Creation Status Window: About to Create...(DB2 Server)	18
37.	Sample Database Creation Status Window: Creation Completed (DB2 Server)	18
38.	IBM DATABASE 2 - Icon View Window (DB2 Server)	19
39.	Database Director - Tree View Window	19
40.	DB2 - Configure Notebook	20
41.	DB2 - Configure—Protocols Page	20
42.	Information Window	21
43.	Multi - Protocol Transport Services—Logo Window	21
44.	Multi - Protocol Transport Services Window	22
45.	Configure Window	22
46.	LAPS Configuration Window	23
47.	Parameters of IBM OS/2 NETBIOS Window	24
48.	CONFIG.SYS Updated Window	24
49.	Exiting MPTS Window	25

50.	OS/2 Warp Connect Install - Installation Drive Window	26
51.	OS/2 Warp Connect Install - Copying files Window	27
52.	Local versus Remote Window	27
53.	Installing OS/2 Warp Connect Window	28
54.	Product Selection Window	28
55.	Set Up Selected Products Window—Adapter Options	29
56.	Set Up Selected Products Window—LAN Requester	30
57.	The Setup Is Complete Window	31
58.	Desktop Window	31
59.	Install Window (IBM DB2 for OS/2 - Single-User)	32
60.	Install - directories Window (DB2 Single-User)	32
61.	Installation and Maintenance Window (DB2 Single-User)	33
62.	Using DB2 for the First Time Window (DB2 Single-User)	33
63.	Local Logon Window (DB2 Single-User)	34
64.	Target Drive Selection Window (DB2 Single-User)	34
65.	Sample Database Creation Status Window: About to Create (DB2 Single-User)	34
66.	Sample Database Creation Status Window: Creation Completed (DB2 Single-User)	35
67.	IBM DATABASE 2 Software Developer 's Kit for OS/2 Installation Window	35
68.	Install Window (IBM DB2 Software Developer 's Kit for OS/2)	36
69.	Install - Directories Window (DB2 SDK)	36
70.	Install - Progress Window (DB2 SDK)	37
71.	Installation and Maintenance Window (DB2 SDK)	37
72.	DB2 Client Setup - Product Information Window	38
73.	DB2 Client Setup Window (before adding node)	38
74.	New Node Window	39
75.	DB2 Client Setup Window (after adding node)	39
76.	DB2 Client Setup - Databases Window	40
77.	New Database Window	41
78.	Client Application Enabler/2 Window	41
79.	IBM DATABASE 2 - Icon View Window (DB2 Client)	42
80.	Command Line Processor Window	42
81.	Installation Window (IBM VisualAge for COBOL, Standard for OS/2)	44
82.	Install Window (VisualAge for COBOL for OS/2)	44
83.	Install - directories Window (VisualAge for COBOL)	45
84.	Install - Progress Window (VisualAge for COBOL)	45
85.	VisualAge COBOL Installation Window	46
86.	SMARTsort 1.1.0 Defaults Window	46
87.	VisualAge COBOL Install Window	47
88.	Installation and Maintenance Window (VisualAge for COBOL)	47
89.	VisualAge for COBOL—Icon View Window	48
90.	Templates - Icon View Window	48
91.	COBOL Project - Settings Window: Target Page	49
92.	COBOL Project - Settings Window: Location Page (Source Directories)	49
93.	Create Directories Window	50
94.	COBOL Project - Settings Window: Location Page (Working Directory)	50
95.	COBOL Project - Settings Window: Inheritance Page	51
96.	Select a Project to Inherit from Window	51
97.	SALESDEP - Settings General Page	52
98.	SALESDEP - Icon View Window (before building the project)	53
99.	GUI Compile: File Scope—IBM COBOL Compiler Options Window	53
100.	GUI Compile: File Scope - IBM COBOL Compiler Options Window—Execution Options	54

101.	GUI Compile: File Scope - IBM COBOL Compiler Options Window—Link Options	55
102.	Edit SQL Coprocessor Options Window	55
103.	GUI Compile: File Scope - IBM COBOL Compiler Options Window—Subsystem Options	56
104.	Node Logon Window	57
105.	SALESDEP - Icon View Window (after building the project)	57
106.	New Salary for Sales Department Window	58
107.	New Salary for Sales Department Window: Original and New Salaries	58
108.	OS/2 LAN-Based Environment Including CICS	59
109.	Installation Window: CICS for OS/2	61
110.	Install Window with CICS Identification	61
111.	Install - Directories Window	62
112.	Install - Progress window	62
113.	Install—System Configuration Window	63
114.	Installation and Maintenance Window Showing Completion	63
115.	CICS for OS/2 Version 3.0 - Icon View Window	63
116.	CICS for OS/2 V3.0 Window, Showing Message Logs	64
117.	CICS for OS/2 Terminal V123 Window	65
118.	CICS Sign-on Panel	66
119.	Resource Definition Online Panel	67
120.	System Initialization Table Panel	67
121.	CICS for OS/2 Terminal V123 Window—System Initialization Table-2 Panel	68
122.	CICS for OS/2 Version 3.0 Window—Log, Showing Completion Message	69
123.	CICS Logo Window for CICS Client for OS/2	71
124.	Instructions Window for CICS Client Installation	71
125.	Install Window for CICS Client for OS/2	71
126.	Install - Directories Window for CICS Client	72
127.	Install—Progress Window for CICS Client	72
128.	IBM CICS Client for OS/2 - Language Selection Window	73
129.	IBM CICS Client for OS/2 - Questions Window	73
130.	Installation and Maintenance Window	73
131.	CICS Terminal Window for Resource Definition Online	75
132.	CICS Terminal Window for Signon Table	76
133.	CICS Terminal Window for Signon Table-1	77
134.	Server - Icon View Window for the CICS Application	79
135.	COBOL Compiler: File scope-IBM COBOL Compiler Options Window—Link Options	80
136.	COBOL Compiler: File scope-IBM COBOL Compiler Options Window—Subsystem Options	80
137.	Sample Project 5—Icon View Window	81
138.	Client GUI-Icon View Window	82
139.	Client GUI-Icon view Window for Compilation	83
140.	Employee Lookup Window	84
141.	Templates-Icon View Window	85
142.	COBOL Project-Settings Window for the Target (Server)	86
143.	COBOL Project-Settings window for the Server Location	86
144.	Create directories Window for the Server	87
145.	COBOL Project-Settings Window for Server (General)	87
146.	EMPLLU Server-Settings Window	88
147.	COBOL Project-Settings Window for the Target Location	88
148.	COBOL Project-Settings Window for General Settings	89
149.	EMPLLU Service Calculation-Settings Window	89
150.	EMPLLU Server-Icon View Window	90

151.	EMPLLU Server-Tools Setup Window	91
152.	Add Environment Variable Window-LIB	91
153.	Add Environment Variable Window—SYSLIB	92
154.	COBOL Compiler: File scope-IBM Compiler Options Window for Syntactical Options	92
155.	COBOL Compiler: File scope-IBM COBOL Compiler Options Window—Link Options	93
156.	COBOL Compiler: File scope—IBM COBOL Compiler Options Window—Subsystem Options	94
157.	COBOL Compiler: File scope-IBM COBOL Compiler Options Window—Other Options	94
158.	COBOL Project-Settings Window for the Target (Client)	96
159.	COBOL Project-Settings Window for Client Location Settings	96
160.	COBOL Project-Settings Window for Client Inheritance	97
161.	Select a project to inherit from Window	98
162.	COBOL Project-Settings Window for General Client Settings	98
163.	EMPLLU Client-Settings Window for Client File Settings	99
164.	EMPLLU Client-Icon view Window	100
165.	Employee Lookup Window	101
166.	Node Logon Window	101
167.	Transaction Assistant - Required CICS ECI Parameters Window	103
168.	Transaction Assistant - Required CICS ECI Parameters Window Filled	104
169.	Transaction Assistant - Optional Fields Window—CICS ECI Parameters	104
170.	Overview of DB2 for AIX client/server environment	110
171.	Input Device/Directory for Software Panel	111
172.	Software to Install Panel	112
173.	Add a Group Panel	113
174.	Add a User Panel	114
175.	Database List	115
176.	Database Information	115
177.	List of Tables with Information	116
178.	Available Network Interfaces	117
179.	Minimum Configuration & Startup Panel	118
180.	OS/2 System - Icon View Window	119
181.	OS/2 Warp Connect Install/Remove - Icon View Window	120
182.	Local versus Remote Window	120
183.	Installing OS/2 Warp Connect Window	121
184.	Product Selection Window	121
185.	Set up Selected Products Window—Adapter Page	122
186.	Set Up Selected Products—TCP/IP Page	122
187.	Set Up Selected Products—TCP/IP Page Example	123
188.	The Setup is Complete Window	123
189.	TCP/IP-Icon View Window	124
190.	TCP/IP Configuration Window—Hostname Page	124
191.	Nameserver Entry Window	125
192.	Response from DB2 for AIX server against Ping Command	125
193.	DB2 Client Setup Window	126
194.	New Node Window	127
195.	DB2 Client Setup Window with New Defined Node	127
196.	DB2 Client Setup-Database Window	128
197.	New Database Window	128
198.	Client Application Enabler/2 Window	129
199.	COBOL with CICS for AIX - CICS Client for OS/2	131
200.	Add a Group Panel	135
201.	Add a User Panel	136

202.	Change/Show User Characteristics of a User—Root	137
203.	Change/Show User Characteristics of a User—CICS	140
204.	Add User—CICS	142
205.	AIX Console Message	144
206.	Add Listener Panel—Define Listener	145
207.	EMPLLU Makefile	146
208.	Manage Resource(s) Panel—Define CICS Resources	147
209.	Add Program Panel—Define CICS Program	148
210.	Add XA Definition Panel	149
211.	CICSCLI.INI Initialization File	151
212.	OS/2 Window—Start CICS Client	152
213.	OS/2 Window—Check CICS Client	153
214.	CICS TERM.EXE Window—CICS Sign-on	154
215.	Employee Lookup Window—EMPLLU Run on the CICS Client	155
216.	Host Development Offload	158
217.	Installation of Communications Manager Window	159
218.	Target Drive Selection Window	160
219.	Communications Manager Setup Window	160
220.	Copying Files Window	161
221.	Open Configurations Window	161
222.	OS/2 Communications Manager Window—Workstation	162
223.	Communications Manager Configuration Definition-CSCBCM2 Window	162
224.	3270 Emulation through Token-ring Window	163
225.	OS/2 Communications Manager Window—Product Files	163
226.	Install Window—Communications Manager/2 for OS/2 Warp	164
227.	Change CONFIG.SYS Window	164
228.	Communications Manager Completion Window	165
229.	Communications Manager/2 - Icon View Window	165
230.	Multi-Protocol Transport Services Window	166
231.	Configure Window—MPTS	166
232.	LAPS Configuration Window	167
233.	Update CONFIG.SYS Window	167
234.	Edit Receive List (MVS/TSO) Window	170
235.	Receive Files from Host (MVS/TSO) Window	170
236.	COBOL Project-Settings Window—Target Page (Host Client)	172
237.	COBOL Project-Settings Window—Location Page (Host Client)	172
238.	COBOL Project-Settings Window—General Page (Host Client)	173
239.	EMPLLU Host Client-Icon View Window	173
240.	EMPLLU Host Client-Tools Setup Window	174
241.	EMPLLU Host Client—Tools Setup Window (Variable View)	174
242.	Add Environment Variable Window (Host Client)	175
243.	COBOL Compiler: File scope - IBM COBOL Compiler Options Window—Link (Host Client)	176
244.	COBOL Compiler: File scope-IBM COBOL Compiler Options Window—Prep (Host Client)	176
245.	OS/2 Window—CICS MAP Command	178
246.	CICS for OS/2 Terminal V123—Resource Definition Online Panel	179
247.	CICS for OS/2 Terminal V123—Program Control Table Panel	180
248.	CICS for OS/2 Terminal V123—Program Control Table Panel-1	180
249.	CICS for OS/2 Terminal V123 Window—EMPL transaction	182
250.	CICS for OS/2 Terminal V123 Window—Employee Lookup Panel	183
251.	CICS for OS/2 V123 Window—Terminal Employee Lookup Panel, List of Employees on OS/2	184
252.	Database Schema view Window	186
253.	SAMPLE-Schema Window	186

254.	Data Structure Mapping view Window	187
255.	EMPLLU-Mapping Window	187
256.	EMPLLU-Mapping Window—Expanded View	188
257.	EMPLLU-Mapping Message Window	188
258.	Send Files to Host (MVS/TSO) Window—C-C-Untitled	190
259.	Send Files to Host (MVS/TSO) Window—Transfer File List	191
260.	Edit Send List (MVS/TSO) Window	191
261.	Send File(s) Status Window	192
262.	OS/2 Communications Manager Window—Save List	192
263.	C-C-3270 Emulator Window—ISPF Command Shell	193
264.	ISPF Primary Option Menu	196
265.	DB2I PRIMARY OPTION MENU	197
266.	DCLGEN Panel	198
267.	Job Control Language for Create BMS Sample	199
268.	Job Control Language to Compile and Link the Client Program	200
269.	Job Control Language to Compile the Service Calculation Routine	202
270.	Job Control Language to Compile, Link, and Bind the Server Program	203
271.	Define Transaction Panel	205
272.	Define Program Panel	206
273.	Define Mapset Panel	207
274.	Install Panel	208
275.	SPUFI Panel	209
276.	Employee Lookup Panel on MVS	210
277.	Employee Lookup Panel—List of Employees on MVS	211
278.	OS/2 Client with MVS Server	214
279.	Open Configuration Window for APPC	216
280.	OS/2 Communications Manager Window for APPC	217
281.	Communications Manager Configuration Definition—CSCBCM2 Window	217
282.	Communications Manager Profile List Window for APPC	218
283.	Token Ring or Other LAN Types DLC Adapter Parameters Window	218
284.	Local Node Characteristics Window	219
285.	Connections List Window	220
286.	Adapter List Window	220
287.	Connection to a Host Window	221
288.	Partner LUs Window	222
289.	OS/2 Communications Manager Window (3270 emulator information)	222
290.	SNA Features List Window	223
291.	Local LU Window	223
292.	Mode Definition Window	224
293.	Communications Manager—Checking Values Window	224
294.	CNOS Statements	225
295.	CICS Client SNA Server Definition	226
296.	VTAM PU/LU Definition	228
297.	APPL Definition	228
298.	LOGMODE Definition	229
299.	CDRSC Definition	229
300.	Define Connection Panel	231
301.	Define Sessions Panel	232
302.	DFHCNV Macro Definition	232
303.	Client Status OS/2 Window	234

Preface

This redbook details the configuration of a client/server environment consisting of the IBM COBOL, DB2, and CICS family of products on OS/2, AIX and MVS. Installation details as well as application development techniques are covered.

The book was written for those interested in the interaction of the above family of products when the applications are written in COBOL. Technical marketing, application development, and local area networking personnel will find this information particularly useful.

Several practical examples are used throughout the book on the various operating systems for illustration.

Knowledge of COBOL programming, CICS, DB2, OS/2, and AIX is presumed. Application development knowledge in an MVS environment is a prerequisite for the MVS portions of the book.

How This Redbook Is Organized

This redbook contains 266 pages. It is organized as follows:

- Chapter 1, "OS/2 Local Area Network with DB2"
Installation of various local area network products is detailed.
- Chapter 2, "OS/2 LAN with CICS and DB2"
This chapter adds CICS to the client/server scenario.
- Chapter 3, "COBOL with DB2 for AIX and DB2 Client"
The client/server environment is expanded to include an AIX DB2 server machine.
- Chapter 4, "COBOL with CICS for AIX and CICS Client for OS/2"
This chapter adds CICS on AIX to the client/server scenario.
- Chapter 5, "Host Development Offload"
When in an MVS environment, COBOL applications can be transferred to the OS/2 workstation for maintenance or new COBOL applications can be generated on the OS/w workstation and transferred to the MVS host. Both of these scenarios are explained in this chapter.
- Chapter 6, "OS/2 Client and MVS Server"
This chapter explains using the MVS host as a COBOL application server with an OS/2 client interface.

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization San Jose Center.

Andrea Conzett, IBM Switzerland.

Jill Ku, IBM Taiwan.

Ute Lotz, IBM Germany

Thanks to the following people for their invaluable contributions to this project:

Derek Carter
Santa Teresa Laboratory

Dennis Virginia
International Technical Support Organization, Poughkeepsie Center

Hugh Smith
International Technical Support Organization, San Jose Center

This project was designed and managed by:

Joe DeCarlo
International Technical Support Organization, San Jose Center

Comments Welcome

We want our redbooks to be as helpful as possible. Should you have any comments about this or other redbooks, please send us a note at the following address:

redbook@vnet.ibm.com

Your comments are important to us!

Chapter 1. OS/2 Local Area Network with DB2

This chapter describes how to set up an OS/2 LAN-based client/server environment that includes the following products:

- OS/2 Warp Connect V3
- OS/2 LAN Server V4.0
- DB2 for OS/2 Server V2.1.1
- DB2 for OS/2 Single-User V2.1.1
- IBM VisualAge for COBOL for OS/2 V1.2.

Figure 1 shows an overview of the environment.

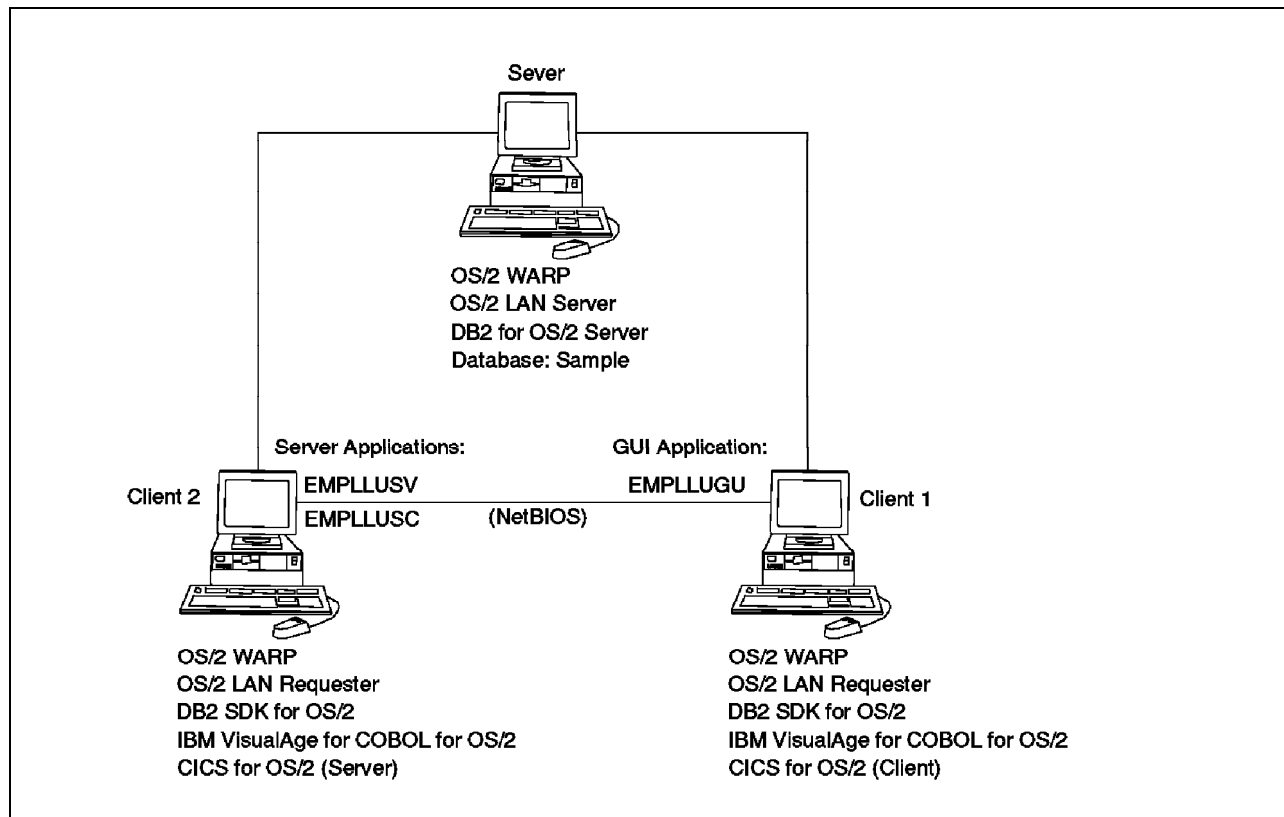


Figure 1. OS/2 LAN-Based Environment

This is just one example of how to install the products on different machines. The environment need not look like that in Figure 1. Therefore we add the following comments:

LAN Server

Your OS/2 workstations may already be connected to an OS/2 local area network (LAN). If so, you do not need to install the OS/2 LAN Server again.

You may, however, want to install the server on one of your workstations in order to define a domain you control. You could then use the hard disk of the server machine to store common COBOL copy books and share them among your developers.

DB2 Server

You can install DB2 for OS/2 Server on any workstation in your LAN; it does not have to be the workstation where the OS/2 LAN Server is installed. However, if you do not have the LAN Server Administration Utilities on your DB2 Server workstation, you must use the User Profile Management Services to define the user IDs allowed to connect to the server databases.

Client Workstations

We use the client workstations not only as requesters in a client/server execution environment (SALESDEP application) but also as application development workstations. However, the application development environment as we set it up is also a client/server environment because the DB2 precompiler is requesting the database definitions from the DB2 Server.

If you use a client workstation to execute a client application only, then you need not install DB2 Software Developer's Kit for OS/2 or IBM VisualAge for COBOL for OS/2. All you would need in terms of DB2 is the DB2 Client Application Enabler for OS/2.

DB2 Software Developer's Kit

You need DB2 Software Developer's Kit for OS/2 installed on your client workstation only if you want to develop DB2 applications as well as execute them. The Software Developer's Kit for OS/2 includes DB2 Client Application Enabler for OS/2 so that you can access the DB2 server databases when you precompile a DB2 program.

DB2 for OS/2 Single-User

DB2 for OS/2 - Single-User provides the basic database functions and can be used for the local database access during the application development process. However, it cannot share its data with other workstations in the LAN. It includes the DB2 Software Developer's Kit for OS/2 as well as the DB2 Client Application Enabler for OS/2.

VisualAge for COBOL

IBM VisualAge for COBOL for OS/2 has to be installed on every workstation where you want to develop COBOL applications, but it is not needed for the execution of a COBOL program. The Package option of IBM VisualAge for COBOL for OS/2 puts all the COBOL run-time libraries you need for execution in one directory that can be installed on the execution client.

1.1 OS/2 Server Installation

A server shares resources (directories, printers, and serial devices) with other workstations on the LAN. Servers belong to logical groups called *domains*. The first server installed in the domain must be the domain controller. The domain controller manages information about users, groups, and workstations in the domain. An additional server can also share its resources but does not manage the domain.

DB2 for OS/2 is a relational database management system that resides on the OS/2 platform. DB2 for OS/2 Server supports the basic database functions and provides support for local or remote database client access. Remote database clients can communicate with the server through a variety of communication protocols. In our installation we used NetBIOS. The clients do not have to know

the physical location of the database when they access data on the server; access is controlled by the DB2 for OS/2 Server.

1.1.1 Installing OS/2 LAN Server 4.0

Follow these steps to install OS/2 LAN Server 4.0:

1. Insert the LAN Server CD-ROM in the CD drive. At an OS/2 command prompt, type:

```
d:\INSTALL
```

where d is the drive letter of the CD-ROM, and press Enter.

2. Click on **OK** on the Welcome window (Figure 2).

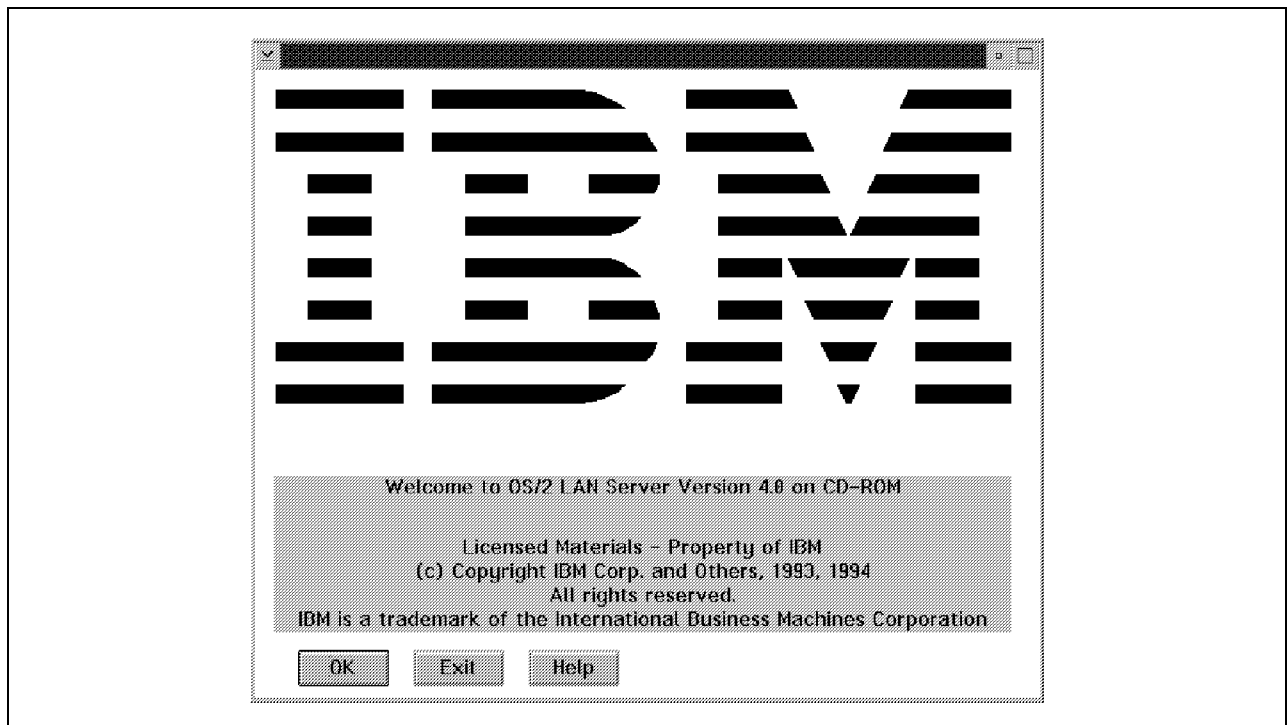


Figure 2. LAN Server Welcome Window

3. Select the **Install OS/2 LAN Server 4.0** option on the CD-ROM Installation window (Figure 3 on page 4) and click on **OK**.

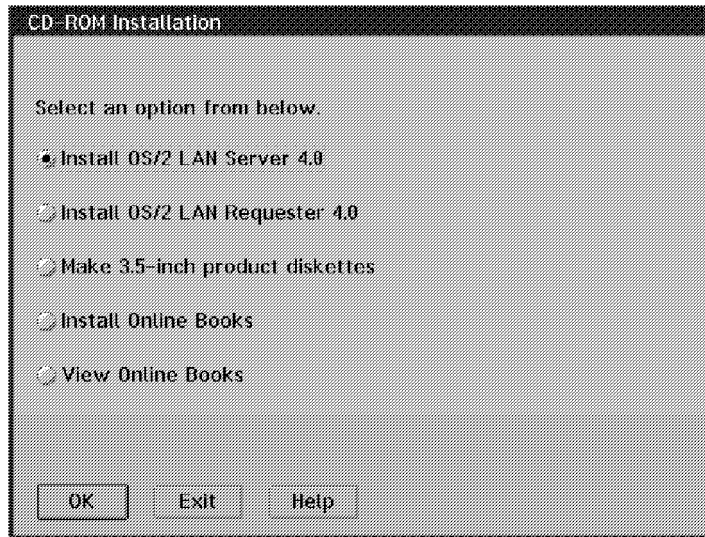


Figure 3. CD-ROM Installation Window

4. Click on **OK** on the Server Installation/Configuration window (Figure 4).

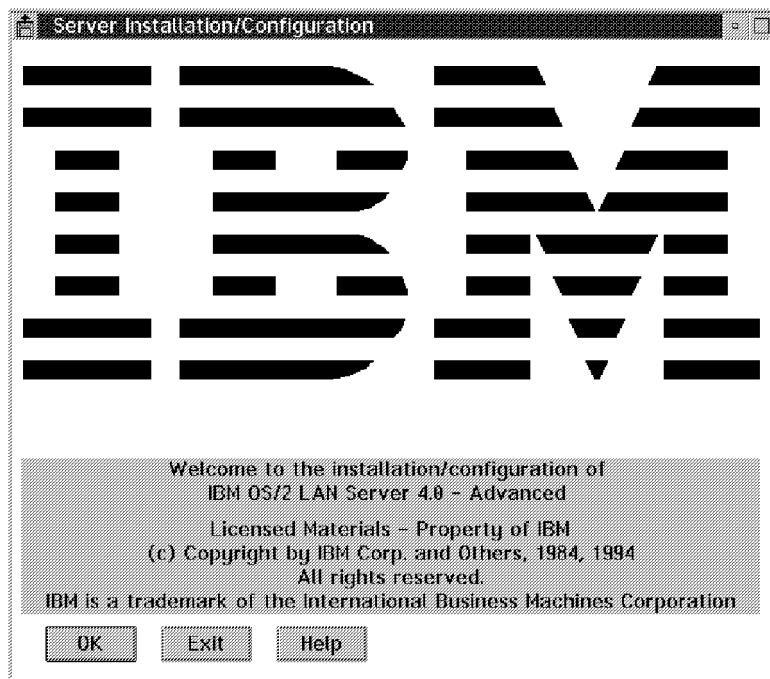


Figure 4. Server Installation/Configuration Window

5. Click on **Easy** on the Easy or Tailored Installation/Configuration window (Figure 5 on page 5).

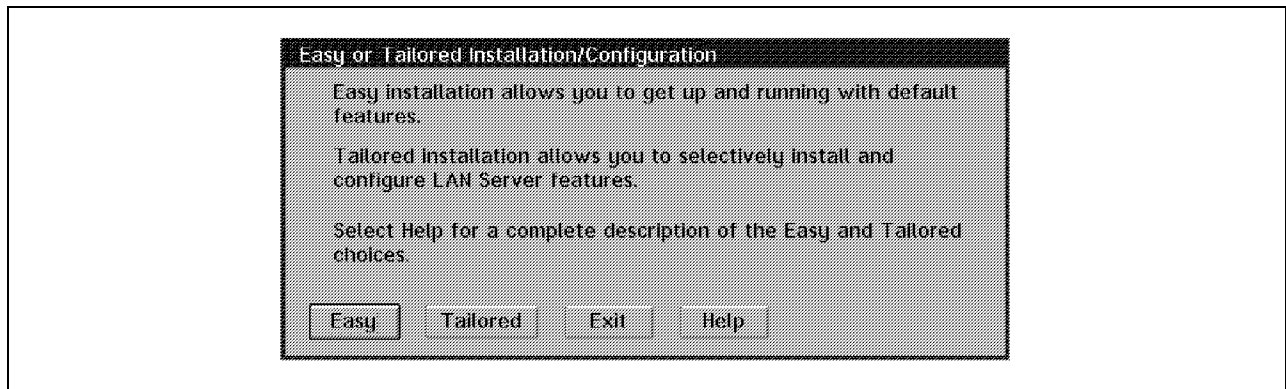


Figure 5. Easy or Tailored Installation/Configuration Window

6. When the Installation Location window is displayed (Figure 6) verify that the correct drive is specified in the *Drive* field. Then click on **OK**.

Note: The LAN Server components that you are about to install require about 20 MB of disk space. Make sure that you have enough space left on the drive specified on the Installation Location window (Figure 6). Also, allow additional disk space for the swapper file.

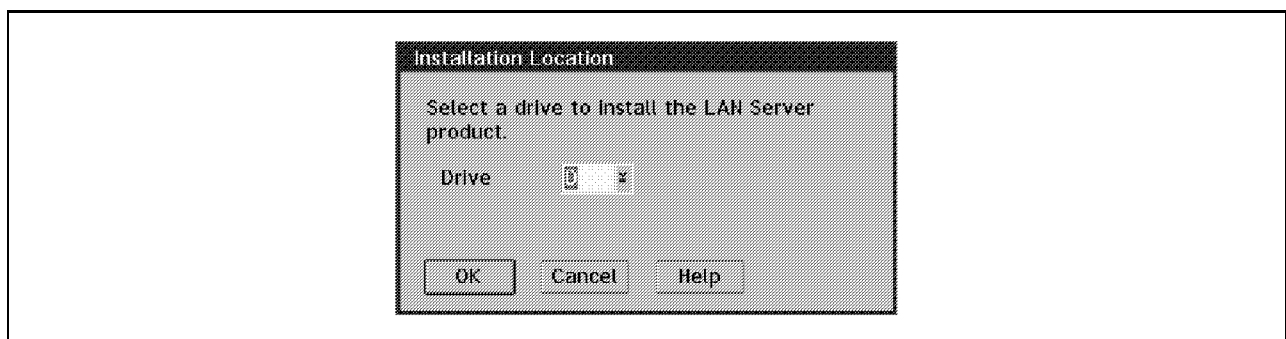


Figure 6. Installation Location Window

7. The Server Name window is displayed in Figure 7. In the *Server* field, type a name for this server and then click on **OK**. The server name must be different from the name of any other server, workstation, or domain on the LAN.

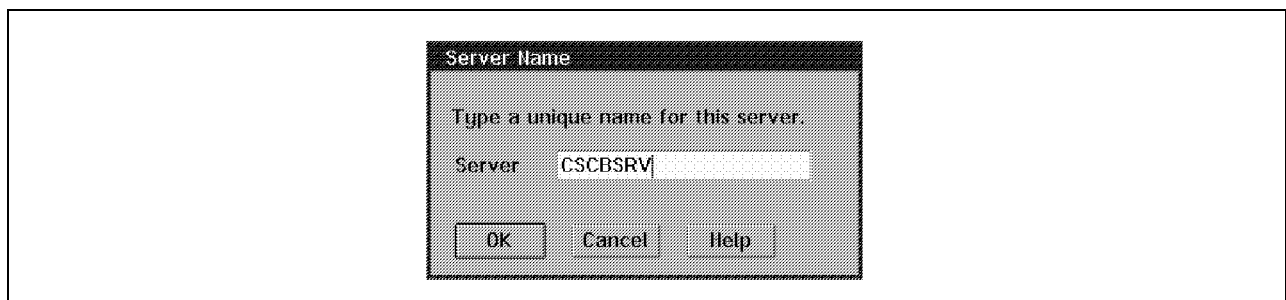


Figure 7. Server Name Window

8. The Domain Name window is displayed in Figure 8 on page 6. In the *Domain* field, type the name of the domain to which the server will belong. Click on **OK**.

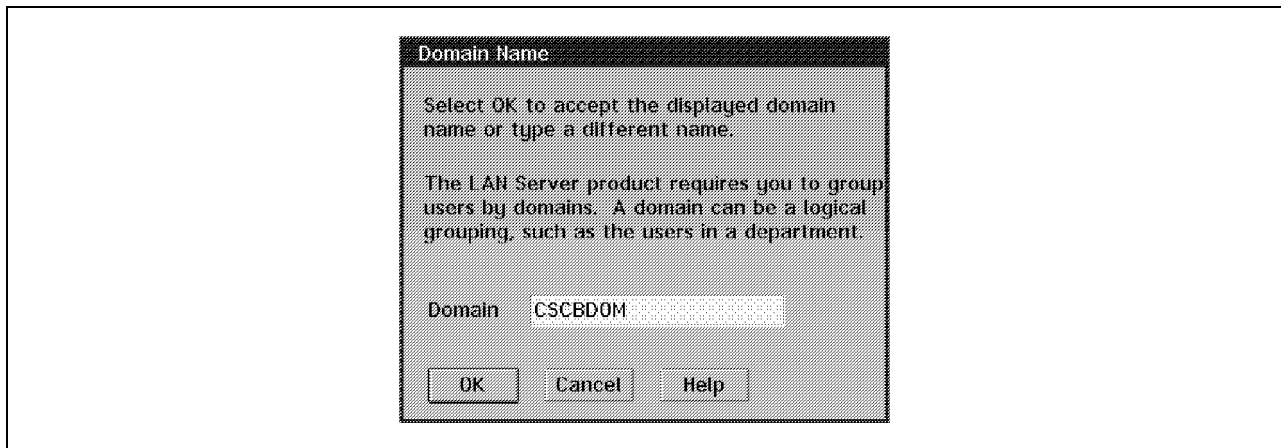


Figure 8. Domain Name Window

9. The First Server—Domain Controller window is displayed in Figure 9 on page 6. Click on **Yes**.

In our installation, we have no additional LAN servers in the domain. Therefore the server that is being installed is the first and only server as well as the domain controller.

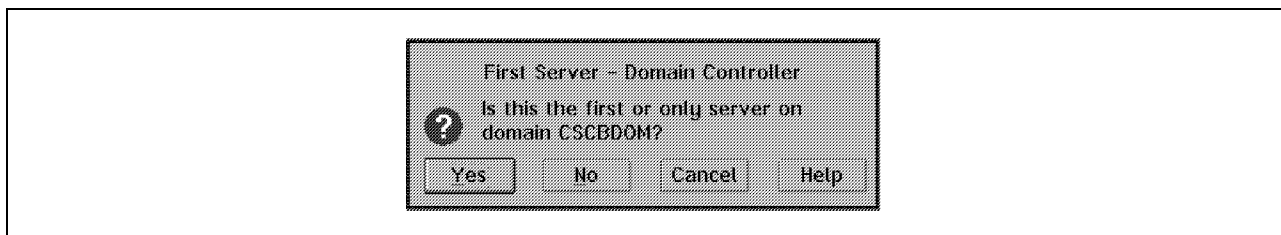


Figure 9. First Server—Domain Controller Window

10. The Network Adapter Confirmation window is shown in Figure 10. Select the appropriate network adapter and click on **OK**.

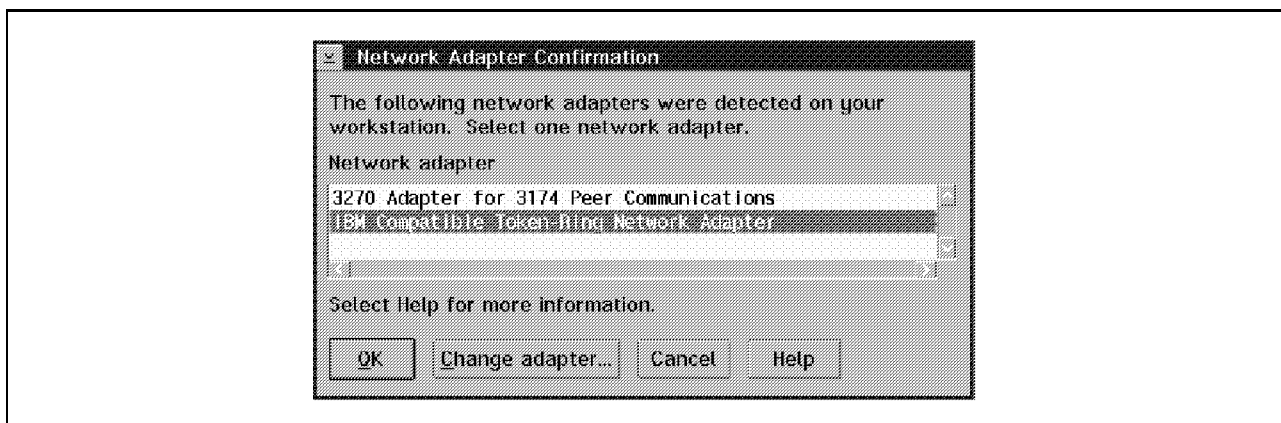


Figure 10. Network Adapter Confirmation Window

The product code is copied from the CD-ROM onto the server workstation's hard disk.

11. After copying the product code to the hard disk, the window Default User ID and Password is displayed (Figure 11 on page 7). Click on **OK**.

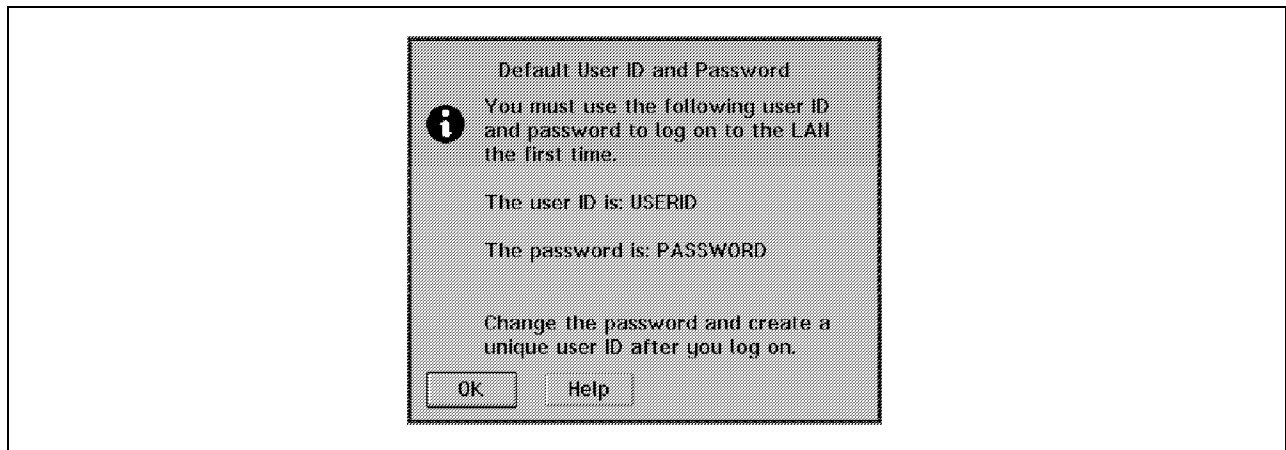


Figure 11. Default User ID and Password Information Window

12. When the window labeled Create a Startup Diskette For Your Workstation appears (Figure 12), click on **Yes**.

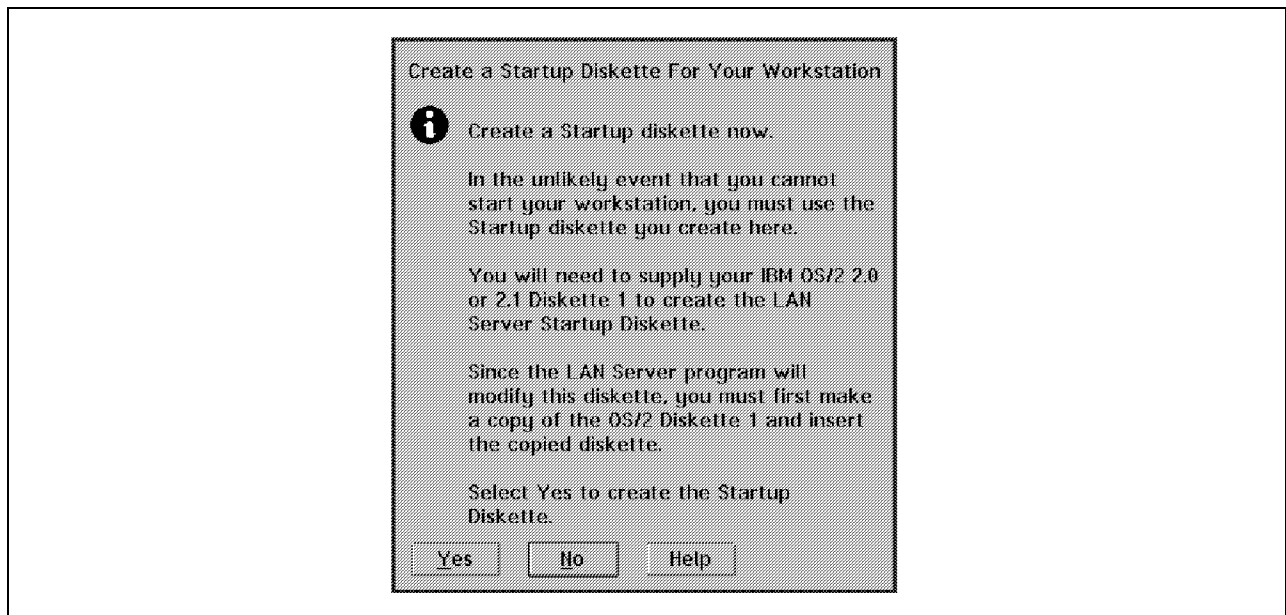


Figure 12. Create a Startup Diskette For Your Workstation Window

13. On the Create a Startup Diskette window (Figure 13 on page 8), specify the diskette drive, insert a copy of the OS/2 Warp Connect Diskette 1 into this drive, and click on **OK**.



Figure 13. Create a Startup Diskette Window

14. A completion message (Figure 14) is displayed when the installation is finished. Click on **OK**.

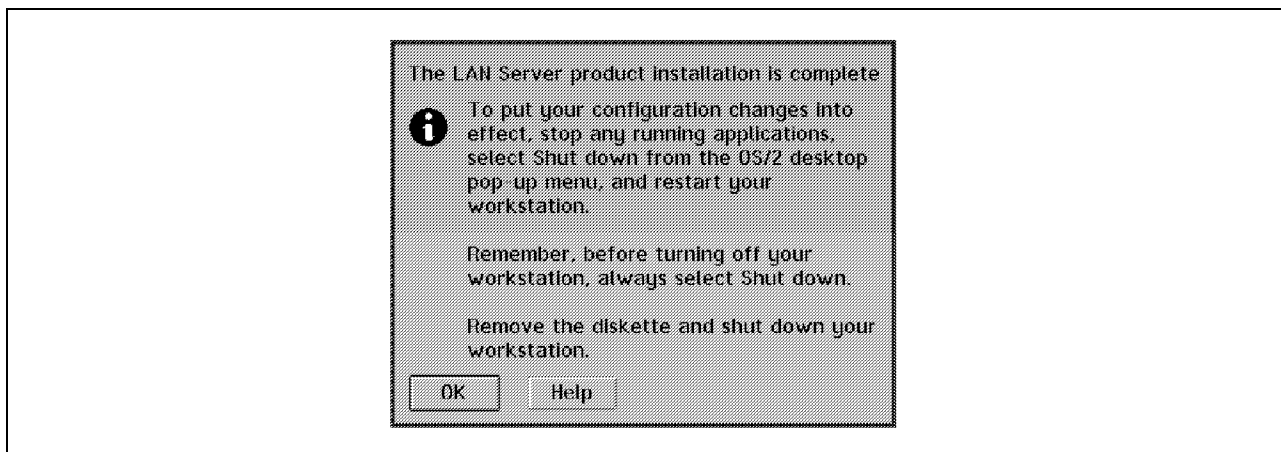


Figure 14. Installation Completion Message Window

Before using the LAN Server product, you must shut down and restart your server workstation.

1.1.2 Defining the Users of the Domain

After successful installation of the LAN Server product and restart of the server workstation, you must define:

- The resources that you want to share
- The users you allow to use these resources
- The privileges these users should have.

USERID is the default administrator ID for both LAN Server and DB2 for OS/2. We recommend changing this ID to have a unique systemwide ID to control the domain. In our example we used CSCBADM.

To define the users of the domain, perform the following steps:

1. On the desktop double-click on the **IBM LAN Services** icon. The IBM LAN Services - Icon View folder will open (Figure 15 on page 9).

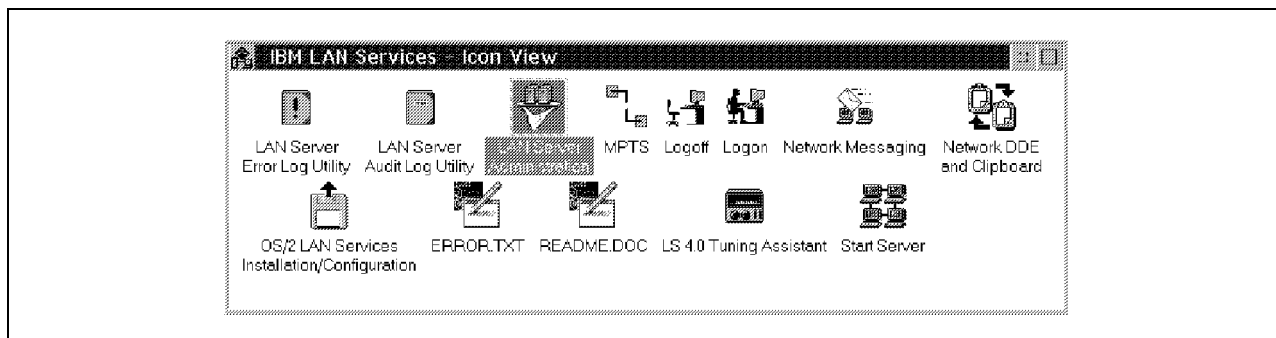


Figure 15. IBM LAN Services—Icon View

2. Double-click on the **LAN Server Administration** icon. The LAN Server Logon window is displayed (Figure 16). Type:

USERID in the *User ID* field

PASSWORD in the *Password* field (the password is not displayed).

The *Domain name* field displays the name you specified when you installed the LAN Server (see Figure 8 on page 6).

Click on **OK**.

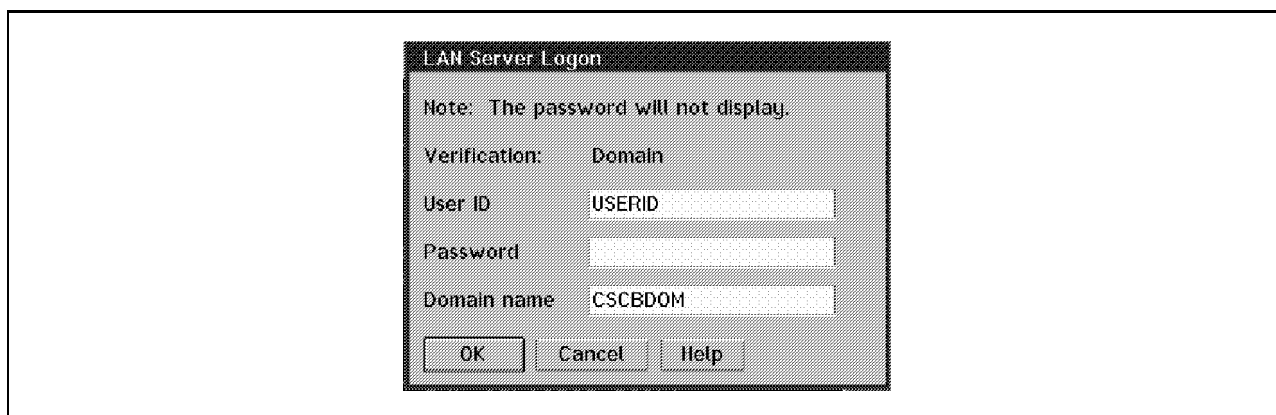


Figure 16. LAN Server Logon Window

Note: Because USERID is the default user ID, it may already have been defined on your network. If so, a message window like that in Figure 17 is displayed. Click on **OK**.

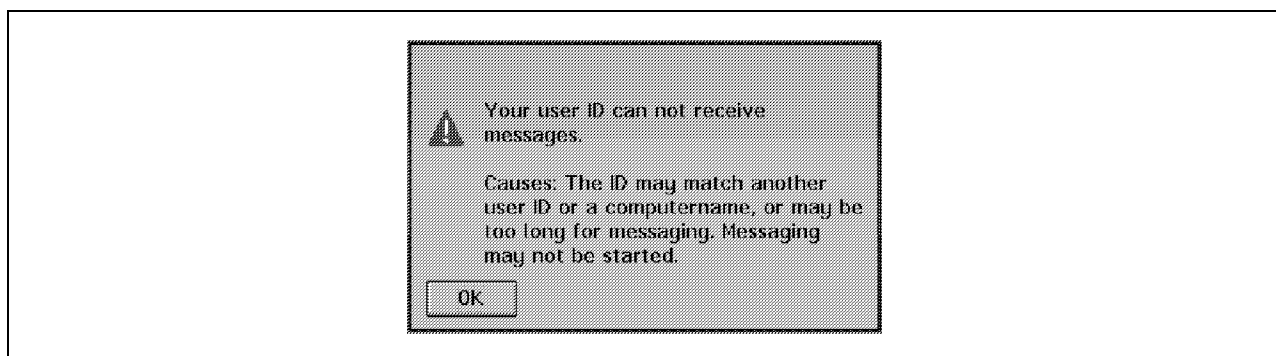


Figure 17. Duplicate ID Message

3. Click on **OK** on the About window (Figure 18 on page 10).

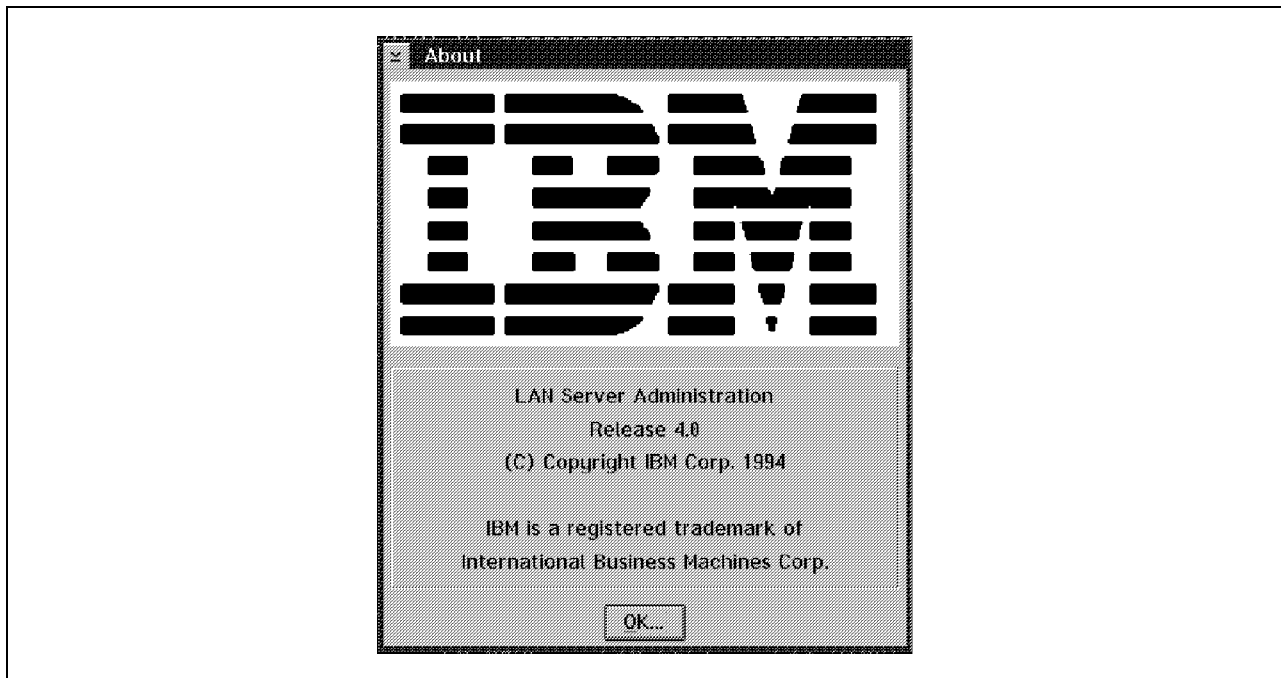


Figure 18. About Window

4. On the LAN Server Administration - Icon View window (Figure 19) open the domain container by double-clicking on the **castle** icon.

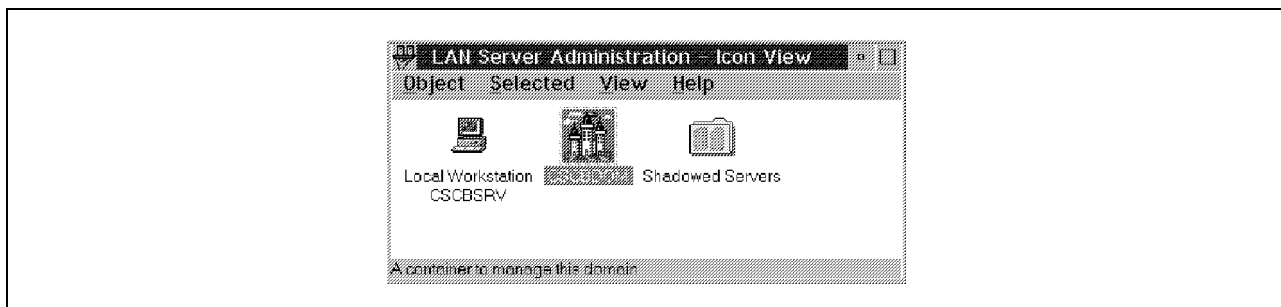


Figure 19. LAN Server Administration - Icon View

5. Double-click on the **User Accounts** icon in the domain folder (Figure 20).

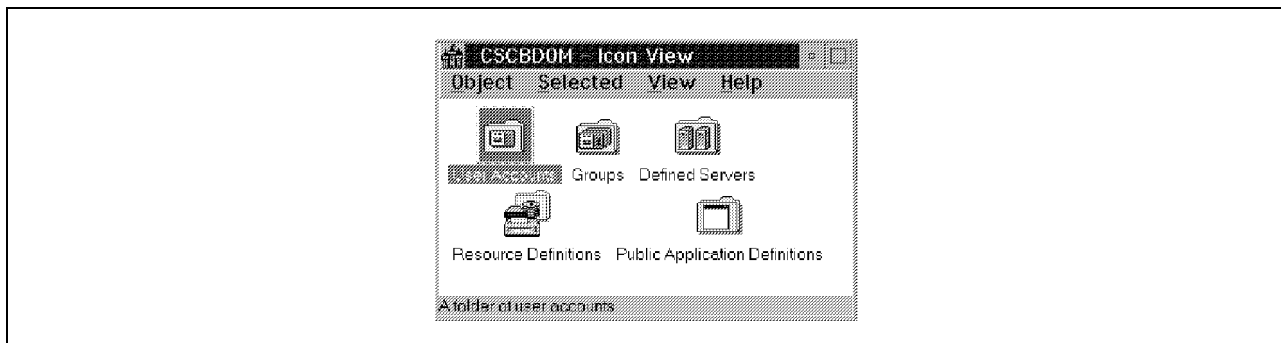


Figure 20. Domain - Icon View

6. On the User Accounts - Icon View (Figure 21 on page 11) move the mouse pointer over the **User ID Template** icon and click mouse button 2. The pop-up menu appears. Select **Create another...** using mouse button 1.

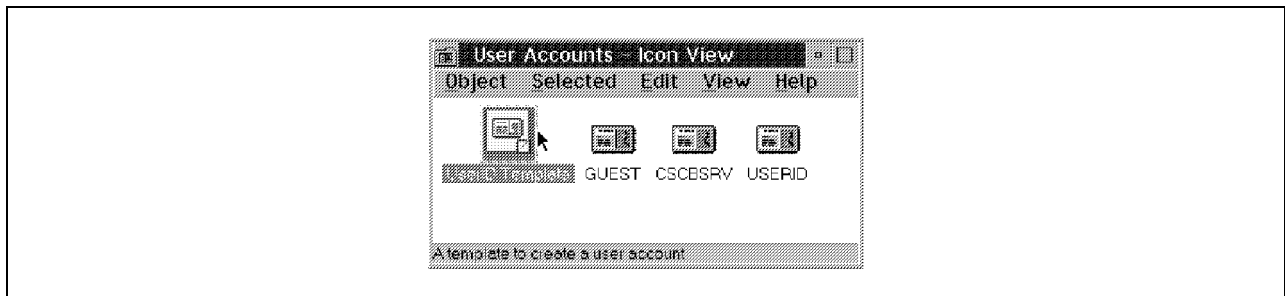


Figure 21. User Accounts - Icon View

7. The User Account-Create notebook appears (Figure 22). In the *User account name* field, type the name of the user ID you want to use as the LAN server administrator ID. The description is optional.

Select the **Password** tab.

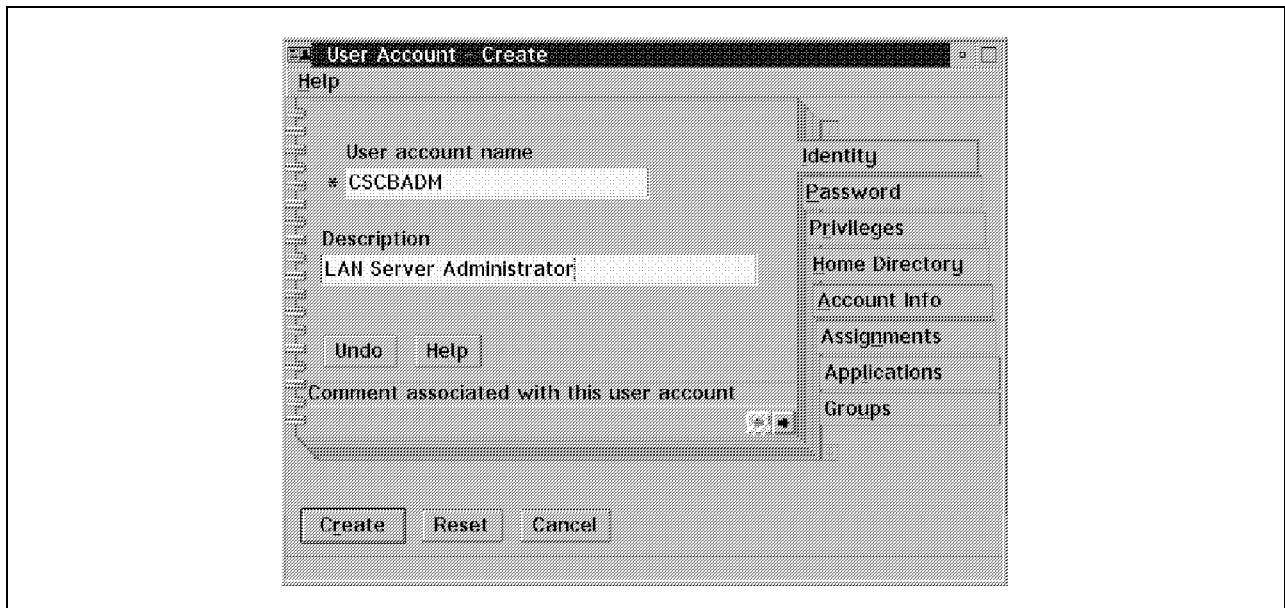


Figure 22. User Account - Create Window—Identity Page

8. Click on the **Change password** check box on the Password page (Figure 23 on page 12) and type a password in the *New password* field and the same password in the *Confirmation* field. The password will be displayed as asterisks.

Select the **Privileges** tab.

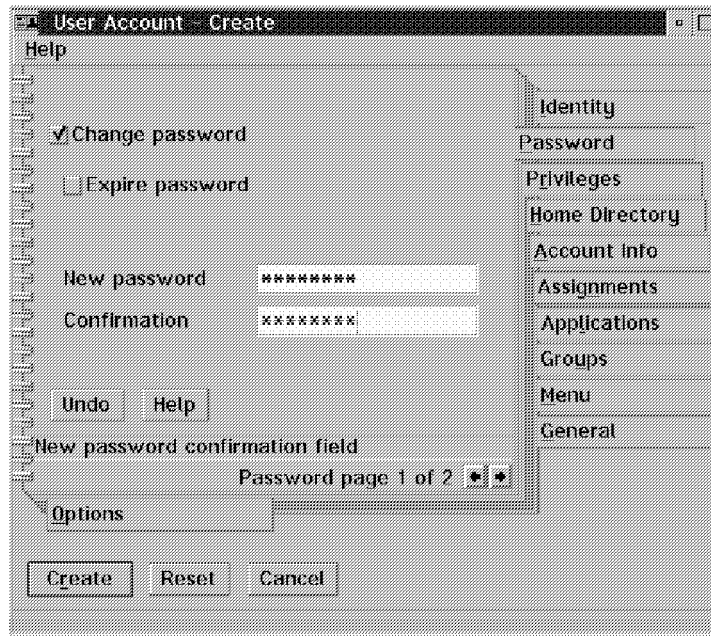


Figure 23. User Account - Create Window—Password Page

9. Select the **Administrator** radio button on the Privileges page (Figure 24) and then click on **Create** in the User Account - Create window.

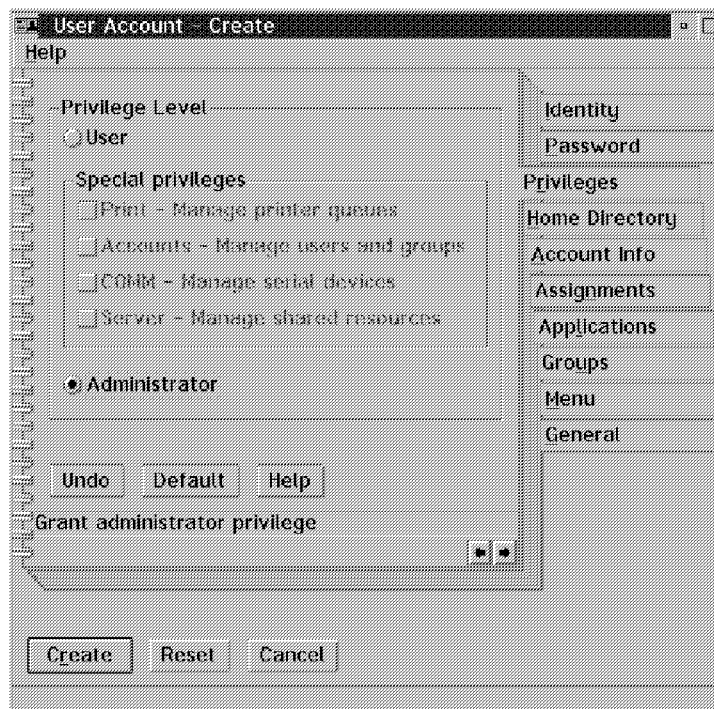


Figure 24. User Account - Create Window—Privileges Page

You have now created a new LAN administrator. To define users that will be able to log on to this domain go through Steps 6 through 9 again. Specify a unique User account name for every user you want to define (such as CSCBU1

and CSCBU2 in our example). On the Privileges page of the User Account-Create (Figure 24) select the **User** radio button instead of the **Administrator** radio button.

Now that you have a new LAN administrator ID, you can delete the default ID. Place the mouse pointer over the **USERID** icon on the User Accounts - Icon View window (see Figure 21 on page 11) and click mouse button 2. When the pop-up menu appears, select **Delete**. The User Account - Delete window will be displayed (Figure 25). Click on **Delete** to confirm the deletion of the default user ID.



Figure 25. User Account - Delete Window

The OS/2 LAN Server is now installed and configured.

1.1.3 Installing IBM DATABASE 2 for OS/2 Server

To install DB2 for OS/2 Server 2.1.1, follow these steps:

1. Insert the DB2 for OS/2 Server CD-ROM in the CD drive. At an OS/2 command prompt, type:
`d:\EN\INSTALL`
where d is the drive letter of the CD-ROM, and press Enter.
2. Click on **OK** on the IBM Database Server window (Figure 26 on page 14).

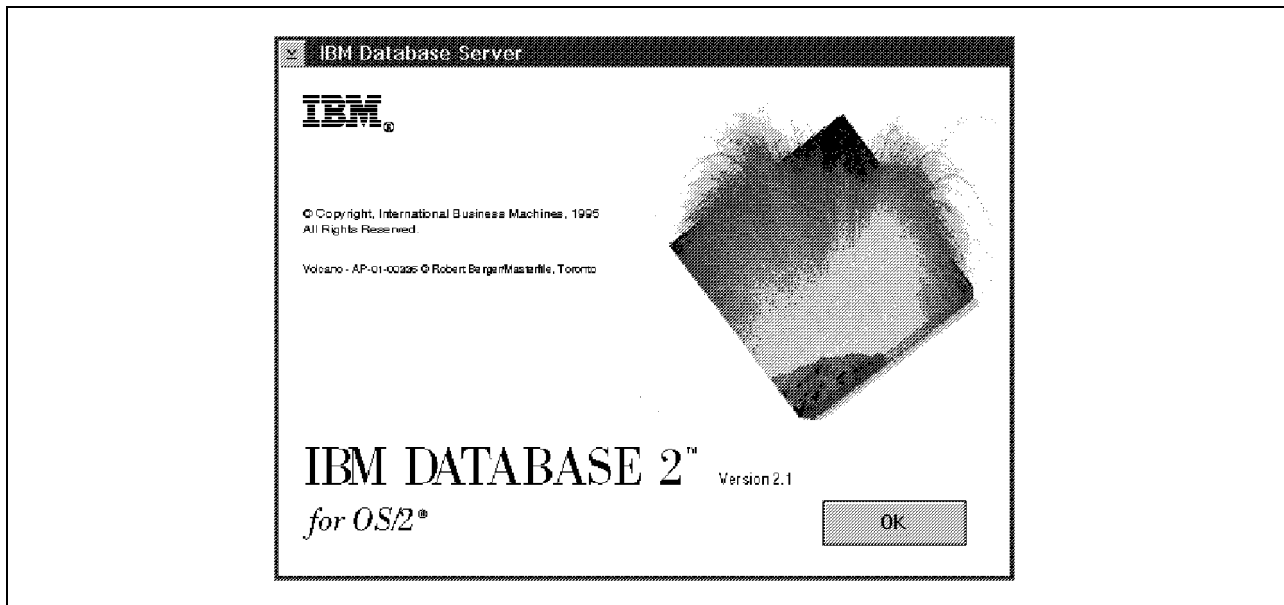


Figure 26. IBM Database Server Window

3. On the IBM Database Server for OS/2—Installation window (Figure 27) ensure that only the **DB2 - Server** check box is selected. Click on the **Software Developer's Kit** and the **DB2 World Wide Web Connection** check boxes to deselect them. Click on **Continue**.

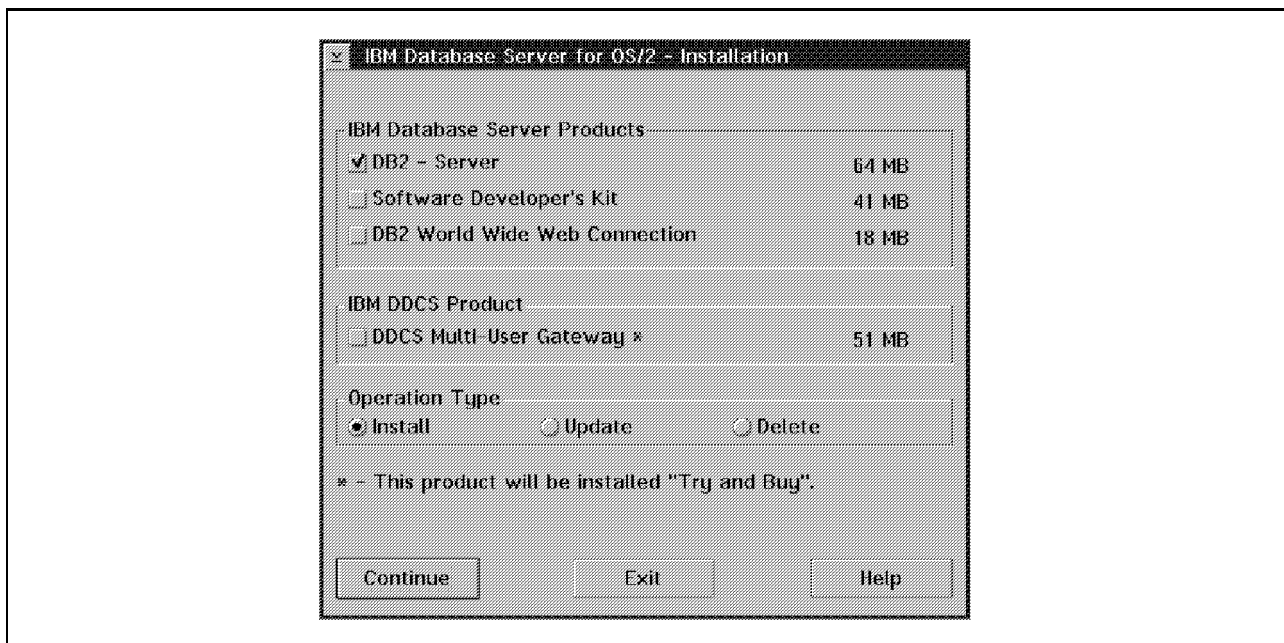


Figure 27. IBM Database Server for OS/2 - Installation Window

4. Ensure that the **Update CONFIG.SYS** check box is checked on the Install window (Figure 28 on page 15). Click on **OK**.

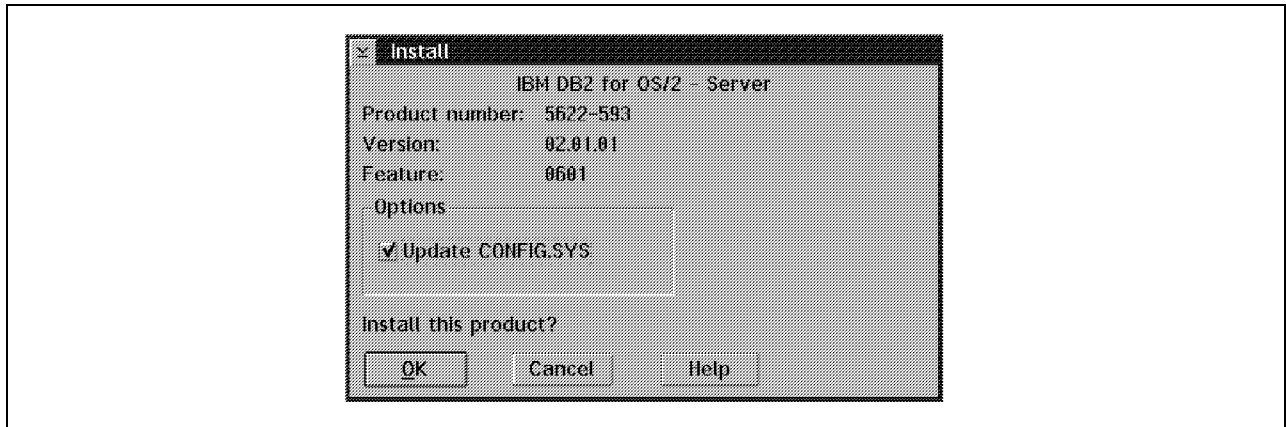


Figure 28. Install Window (DB2 for OS/2 Server)

5. The Install-directories window appears (Figure 29). Select **Server, Database Director, Visualizer Flight**, and **Documentation** from the list box.

In the *File directory* field, specify the disk drive where you have enough space available to install DB2 Server.

Note: The Install - directories window shows how many bytes are needed on the hard disk. The sample database needs 8 MB of disk space in addition to that number.

Click on **Install**.

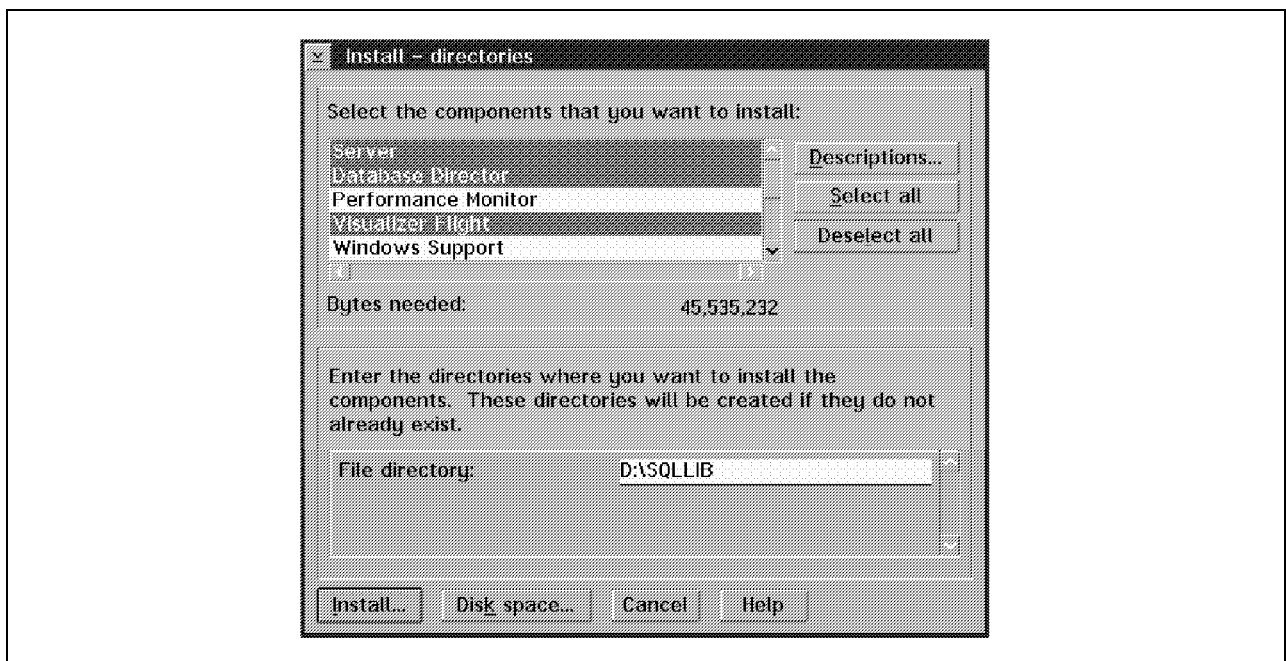


Figure 29. Install - directories Window (DB2 Server)

6. In the *Key* field on the Product License Key window (Figure 30 on page 16), type the license key that is shown on the product key label. Click on **OK**.

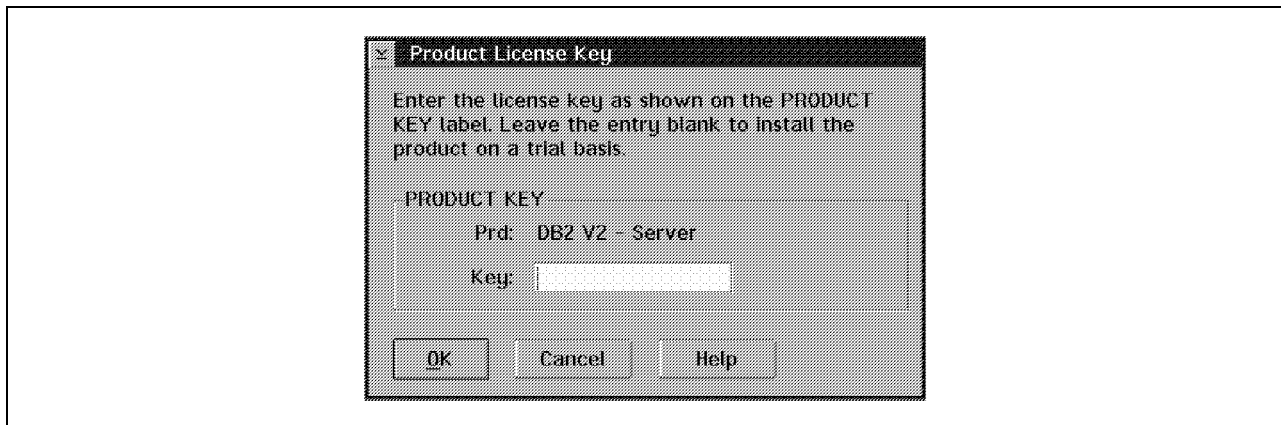


Figure 30. Product License Key Window

The install process is now copying the product code to the hard disk. The Install - progress window (Figure 31) shows the status of the process.

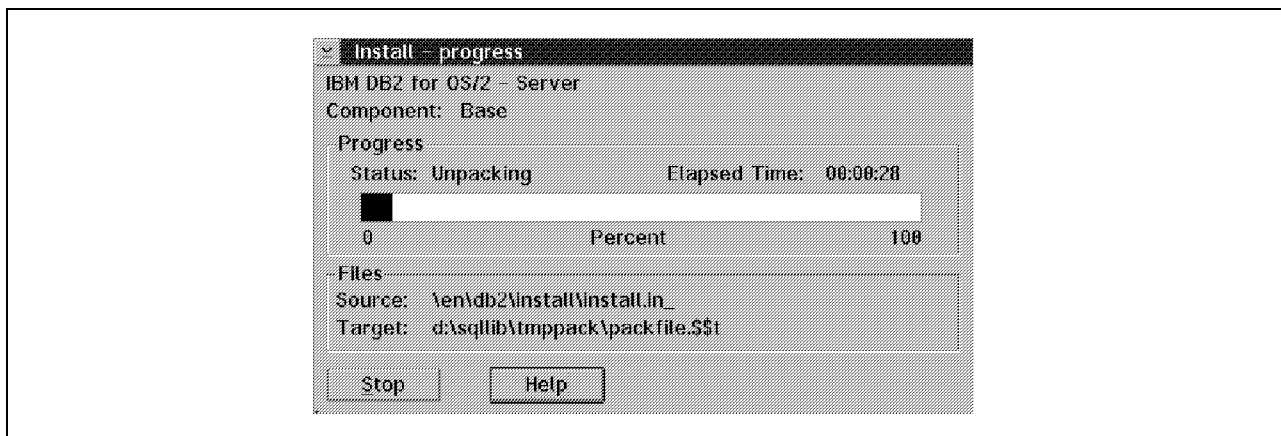


Figure 31. Install - progress Window (DB2 Server)

7. After successful installation, the Instruction ! window appears, as shown in Figure 32. Click on **OK**.

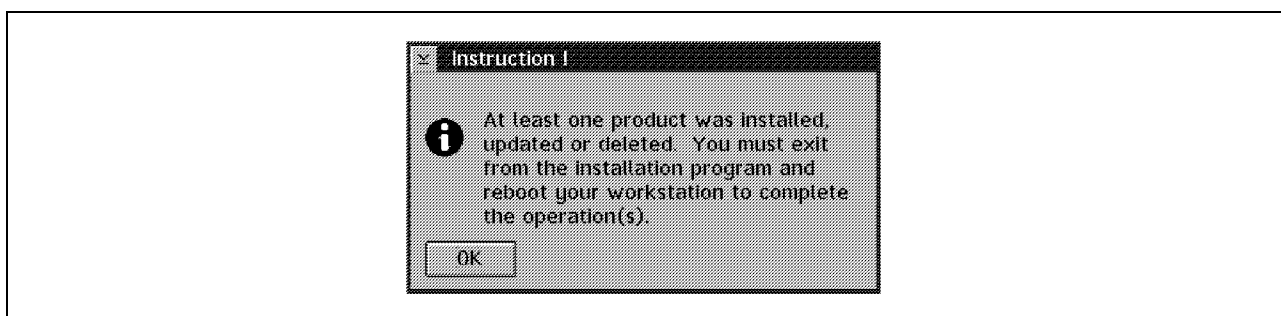


Figure 32. Instruction ! Window

8. Click on **Exit** on the IBM Database Server for OS/2 - Installation window (Figure 27 on page 14). Shut down your workstation and reboot it.

Once you reboot the workstation, the installation of DB2 for OS/2 Server is not yet completed. The last step is the creation of the sample database. Therefore, when the reboot is finished, the window labeled Using DB2 for the first time (Figure 33 on page 17) appears.

To create the sample database:

1. In the window Using DB2 for the first time, click on **Logon dialog...**



Figure 33. Using DB2 for the First Time Window (DB2 Server)

2. When the Local Logon window (Figure 34) appears, type the administrator user ID you defined earlier (CSCBADM in our example) in the *User ID* field and the corresponding password in the *Password* field. Click on **OK**.

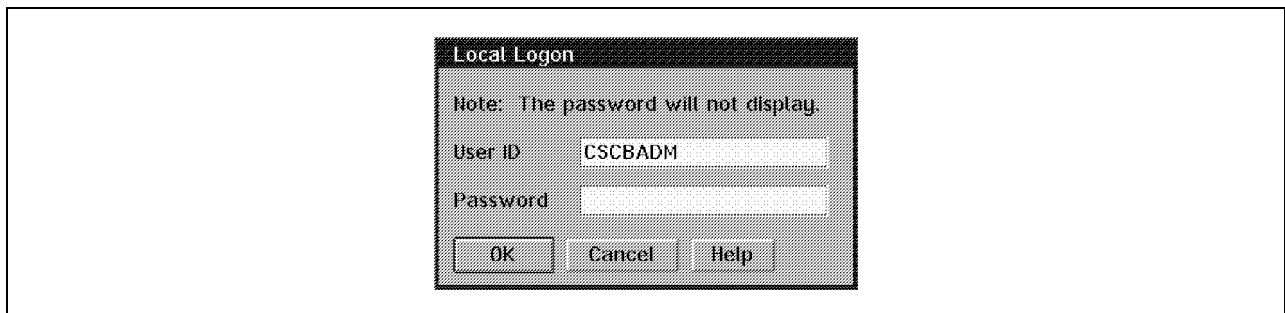


Figure 34. Local Logon Window (DB2 Server)

3. Click on **Create the sample database ...** on the window labeled Using DB2 for the first time (Figure 33).

The window labeled Target drive selection appears (Figure 35 on page 18). Click on **OK** to create the sample database on the same drive where the DB2 for OS/2 Server code is installed.

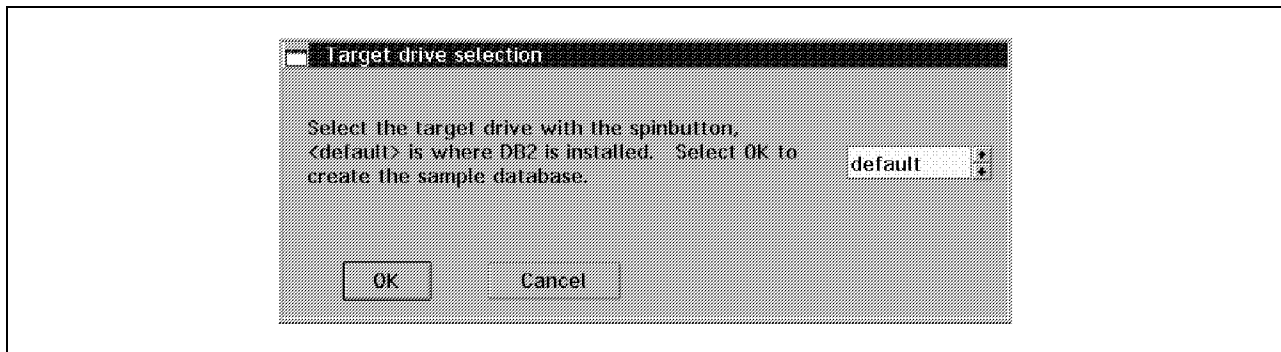


Figure 35. Target Drive Selection Window (DB2 Server)

4. On the window labeled Sample Database Creation Status (Figure 36), click on **OK**.

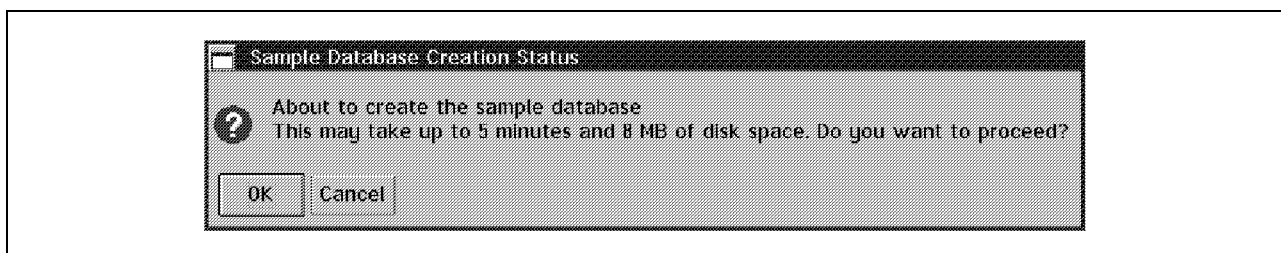


Figure 36. Sample Database Creation Status Window: About to Create...(DB2 Server)

5. After some minutes, a message on the window labeled Sample Database Creation Status (Figure 37) tells you that a sample database with the name **SAMPLE** has been created. Click on **OK**.

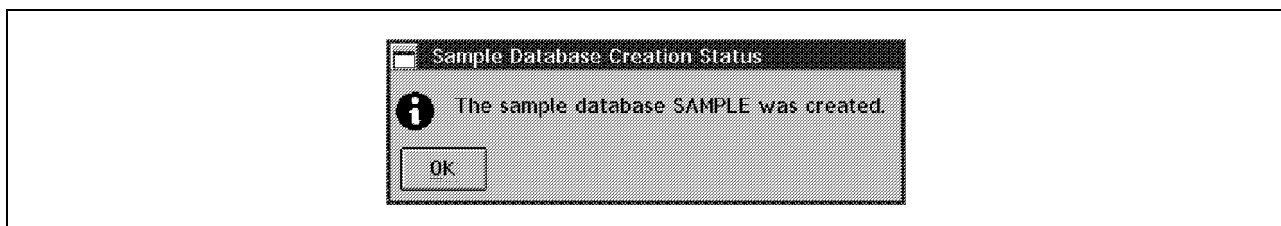


Figure 37. Sample Database Creation Status Window: Creation Completed (DB2 Server)

6. On the window labeled Using DB2 for the first time (Figure 33 on page 17), click on **Cancel** to exit the application.

1.1.4 Configuring DB2 for OS/2 on the Server

If you want to access data on the server from a client workstation, you have to define two parameters:

- The communication protocol
- The node name of the DB2 server workstation.

The server has to be specified as an OS/2 environment variable, stating what communication protocol is used for the data access. In our example we used NetBIOS. Therefore we put the following statement in the CONFIG.SYS file:

```
SET DB2COMM=NETBIOS
```

In a NetBIOS LAN environment, every workstation (server and clients) has to have a unique node name. Client applications must know the node name of the server that contains the database to be accessed. The server's node name must be cataloged in the client node directories. This cataloging can be done through the client setup facility (see 1.2.3, "DB2 Client Setup" on page 37).

The node name of the workstation must be defined in the database manager configuration. To do this:

1. Double-click on the **IBM DATABASE 2** icon on your desktop. The IBM DATABASE 2 - Icon View window will open (Figure 38). Double-click on the **Database Director** icon.

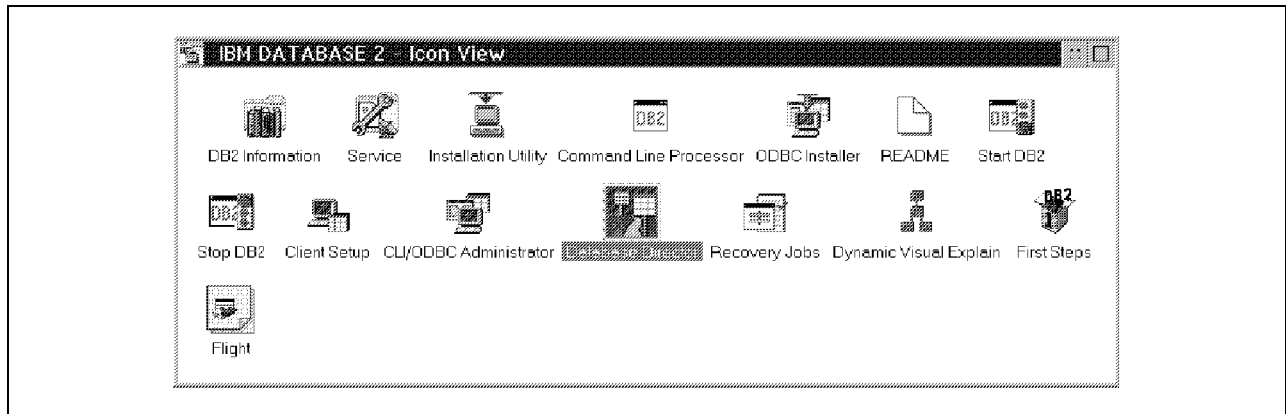


Figure 38. IBM DATABASE 2 - Icon View Window (DB2 Server)

2. On the Database Director - Tree View window move the mouse pointer over the **DB2** icon and click mouse button 1 to select it (Figure 39). Select the **Selected** menu-bar choice on the Database Director - Tree View window, then select **Configure....**

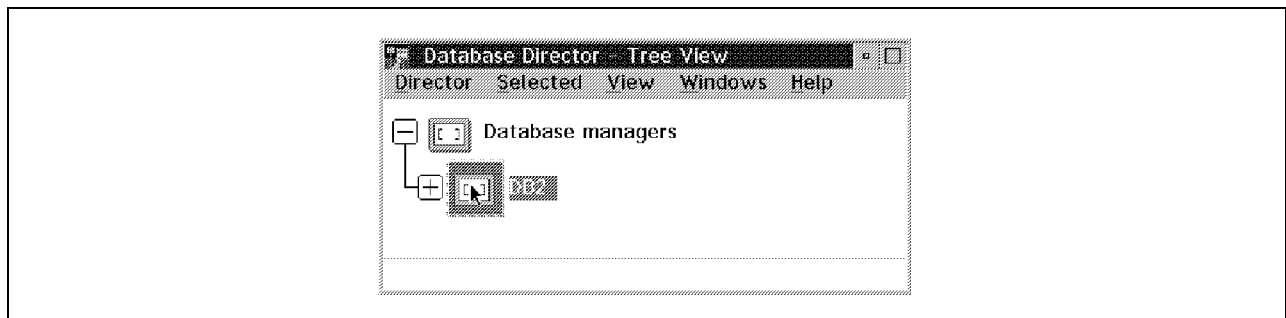


Figure 39. Database Director - Tree View Window

3. The DB2 - Configure notebook appears (Figure 40 on page 20). Select the **Protocols** tab.

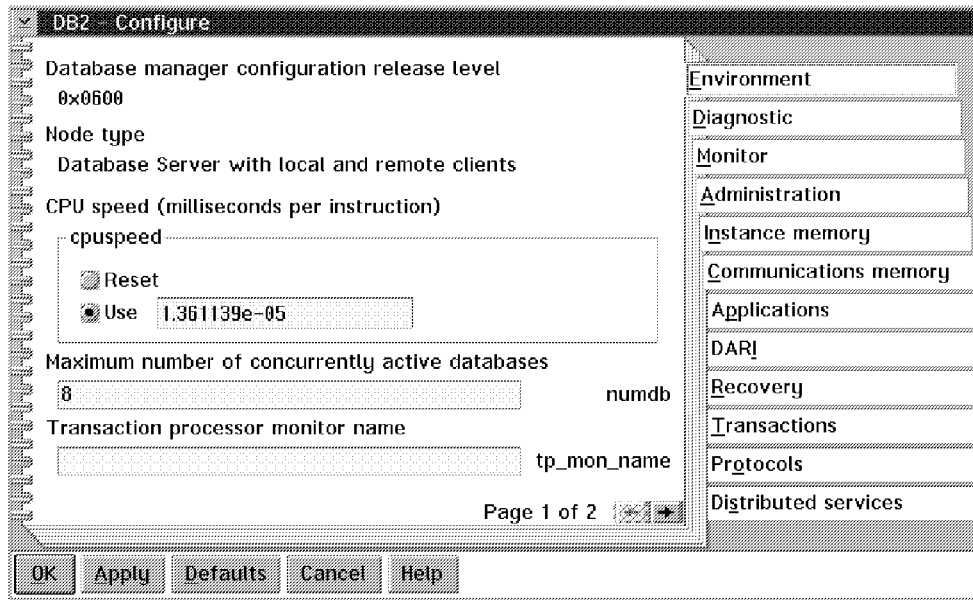


Figure 40. DB2 - Configure Notebook

4. Type a unique node name in the *Workstation name* field on the DB2 - Configure Protocols page (Figure 41). We defined CSCBDBSV as the node name of the workstation where our DB2 server database resides. Click on **OK**.

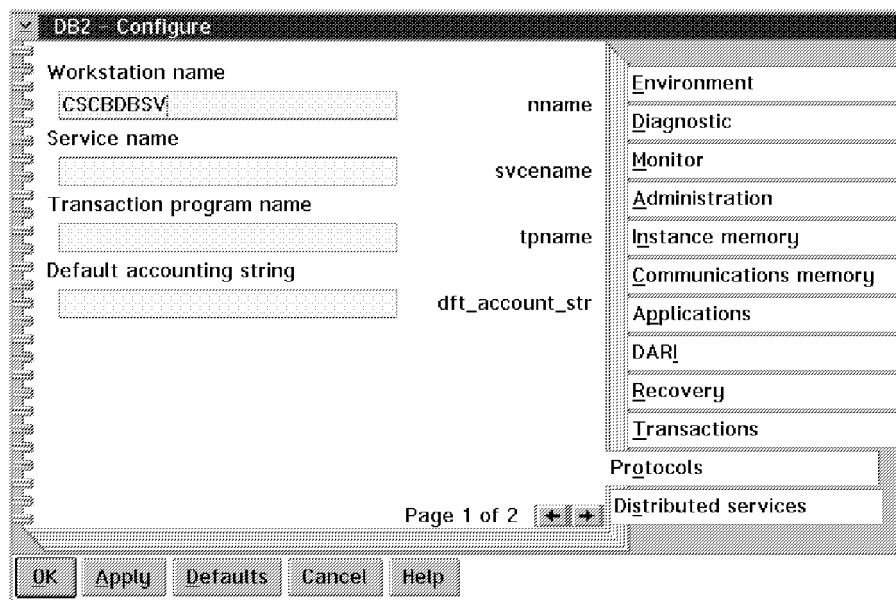


Figure 41. DB2 - Configure—Protocols Page

5. Click on **OK** when the Information window appears as shown in Figure 42 on page 21.

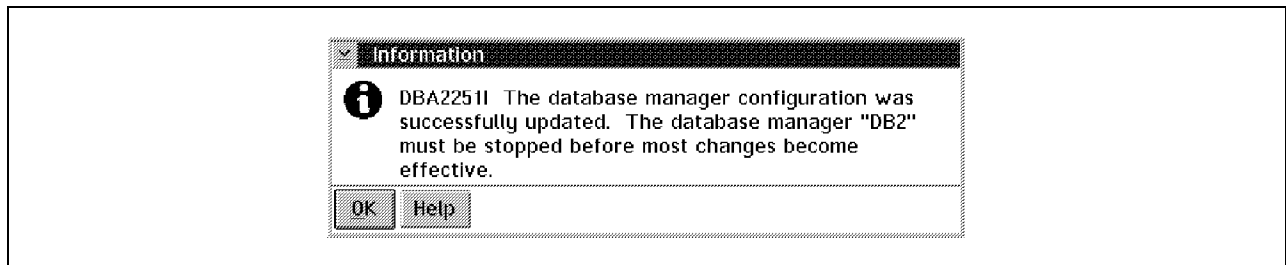


Figure 42. Information Window

Every NetBIOS DB2 client requires NetBIOS resources to be allocated when it initiates a remote database request. The available resources are defined in the NetBIOS system configuration. Therefore you have to change this NetBIOS system configuration on your DB2 server workstation in order to allow one or more DB2 clients to access a server database concurrently.

To do this:

1. Double click on the **MPTS** icon located on your desktop. The Multi-Protocol Transport Services - Logo window appears (Figure 43). Click on **OK** on this window.

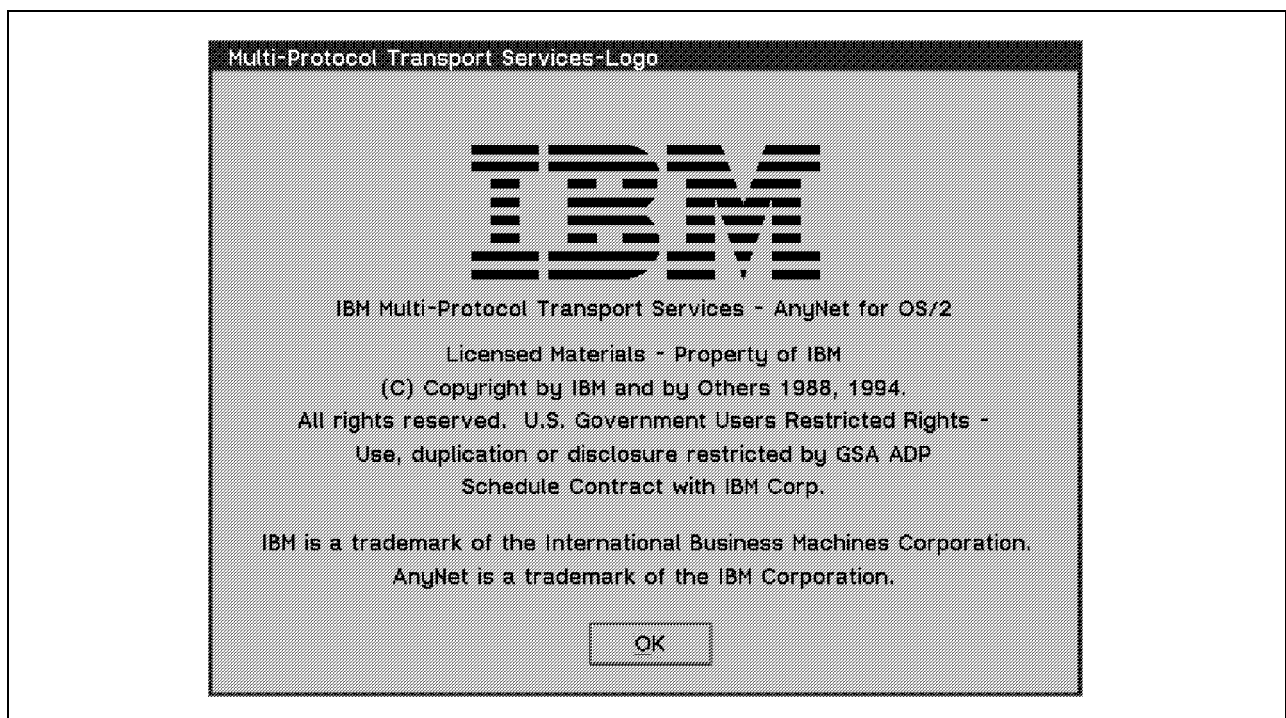


Figure 43. Multi - Protocol Transport Services—Logo Window

2. On the Multi - Protocol Transport Services window click on **Configure** (Figure 44 on page 22).



Figure 44. Multi - Protocol Transport Services Window

3. On the Configure window select **NetBIOS Socket access** and click on **Configure** (Figure 45).

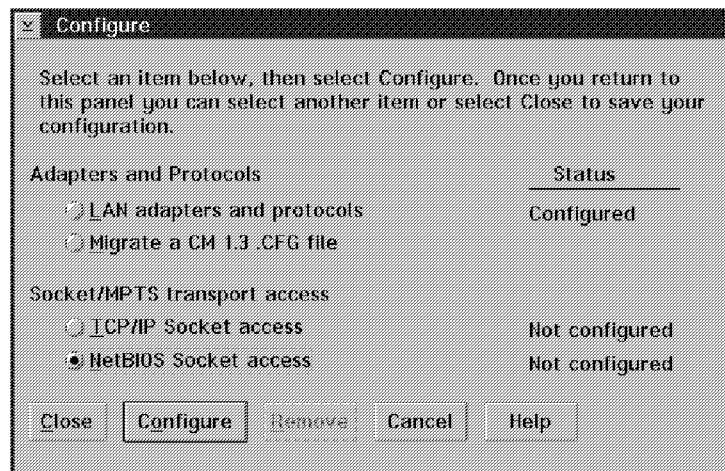


Figure 45. Configure Window

4. The configuration starts, and the Configure window stays open after it has ended. Ensure that only **LAN adapters and protocols** is selected and click again on **Configure**.
The LAPS Configuration window (Figure 46 on page 23) appears. As the current configuration select **IBM OS/2 NETBIOS** and click on **Edit**.

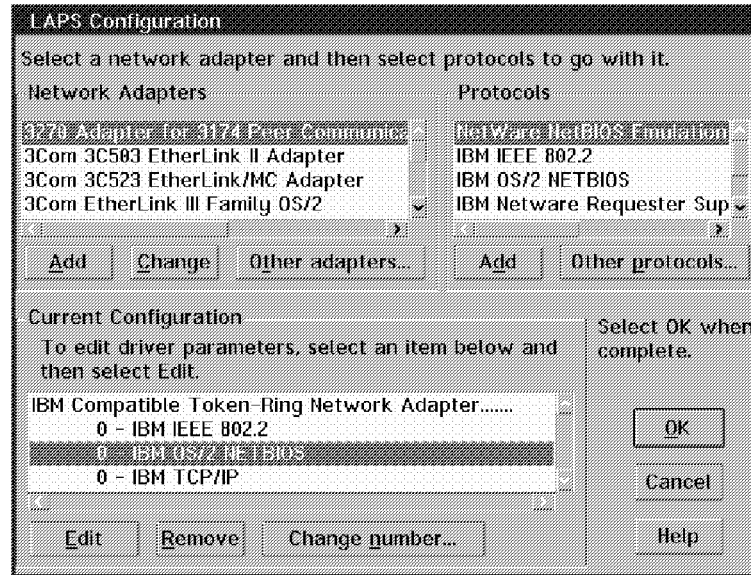


Figure 46. LAPS Configuration Window

5. A window comes up in which you can change all parameters of IBM OS/2 NETBIOS. You have to change three parameters. The number of maximum sessions is given as 130, the maximum number of commands is given as 225, and the maximum names should be 21.

In our environment, we have two DB2 clients that might access the DB2 server concurrently. Therefore, we must add 4 (2 times 2) to the initial settings of these NetBIOS parameters. Doing this sets the parameter for sessions on 134 instead of 130, the parameter for the maximum number on commands on 229 instead of 225, and the parameter for names on 25 instead of 21, as shown in Figure 47 on page 24.

These NETBIOS resources are fixed requirements for each client application running on this workstation.

After the change, click on **OK**. and on **OK** in the LAPS Configuration window.

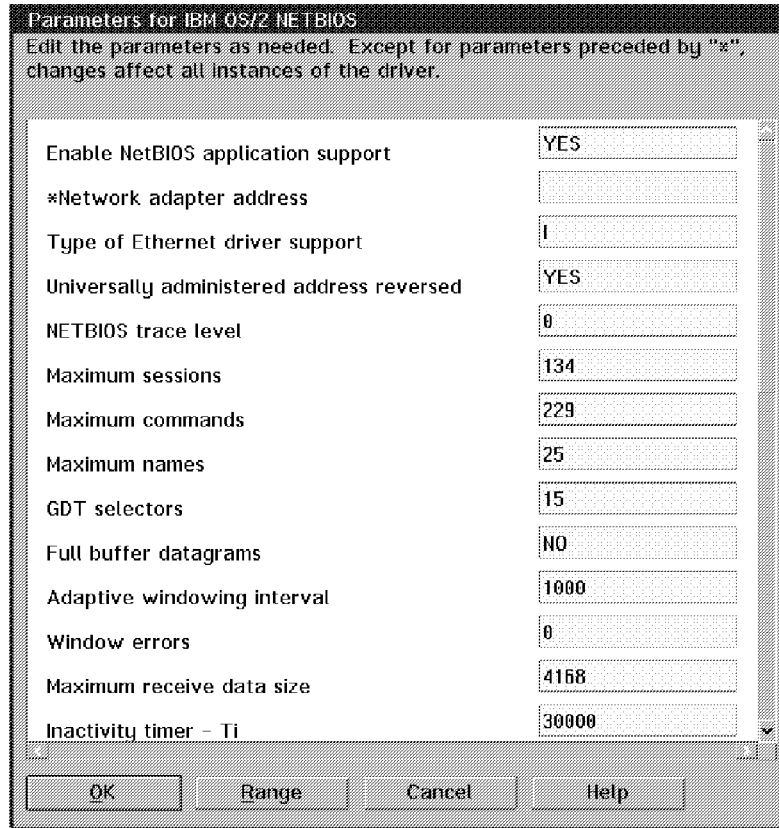


Figure 47. Parameters of IBM OS/2 NETBIOS Window

6. On the CONFIG.SYS Updated window (Figure 48) ensure that **Update CONFIG.SYS** is selected and click on **Exit**.

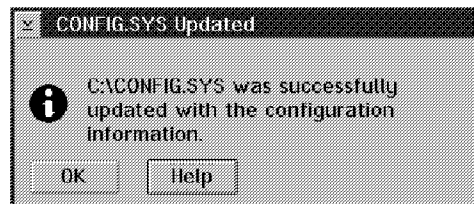


Figure 48. CONFIG.SYS Updated Window

7. After a successful installation, the Exiting MPTS window (Figure 49 on page 25) appears giving you instruction to shut down the system. Click on **Exit**, stop all your running programs, and shut down the computer.

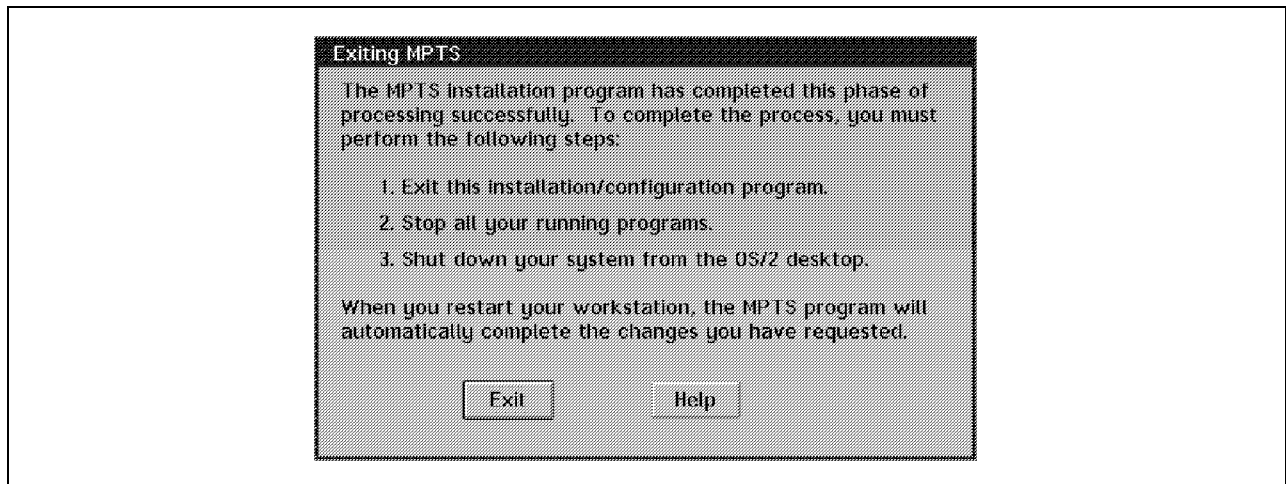


Figure 49. Exiting MPTS Window

NetBIOS Resources

More information about NetBIOS configuration requirements is documented in the publication

DATABASE 2
Installing and Using OS/2 Clients
Version 2

Document Number S20H-4782-01

You find the information you need in the chapter "Configuring the Client Workstation."

You have now installed the products needed on the LAN server:

- OS/2 LAN Server
- and
- DB2 for OS/2 Server

In addition to the installation, identify an administrator who controls the LAN and users who may use the LAN resources. You need to register the server workstation so a user can access DB2 data residing on the server from a client workstation.

In Section 1.2, we describe the installation and configuration of the client workstation from which you may want to access data on the server.

1.2 OS/2 Client Installation

A client or a requester is represented by a workstation from which you can log on to a domain and use the resources that this domain offers. After you log on to a domain, you can access shared resources such as directories and printers and use the processing capability of the servers.

An OS/2 Requester is an OS/2 workstation with the requester functions of the OS/2 LAN Server product installed and running.

OS/2 LAN Requester provides the requester functions on a server workstation or an OS/2 requester workstation.

OS/2 LAN Requester enables you to use shared network resources by sending a request from an application program or user to the network. OS/2 LAN Server receives the request, supplies the shared resource, and passes the response back to OS/2 LAN Requester. The OS/2 LAN Server product does not make you aware of this process. You cannot see the interaction between OS/2 LAN Requester and OS/2 LAN Server.

The OS/2 LAN Server product uses LAN Adapter and Protocol Support LAPS, which is part of MPTS, and LAN Support Program to send data between workstations. The installation program installs the appropriate products on each workstation.

To install LAN Requester 4.0, go through the following steps:

1. Insert the OS/2 CD-ROM into the CD drive. At an OS/2 command prompt, type:

```
d:\INSTALL
```

where d is the drive letter of the CD-ROM, and press Enter.
2. Click on **OK** on the OS/2 Warp Connect Install - Installation Drive window (Figure 50).

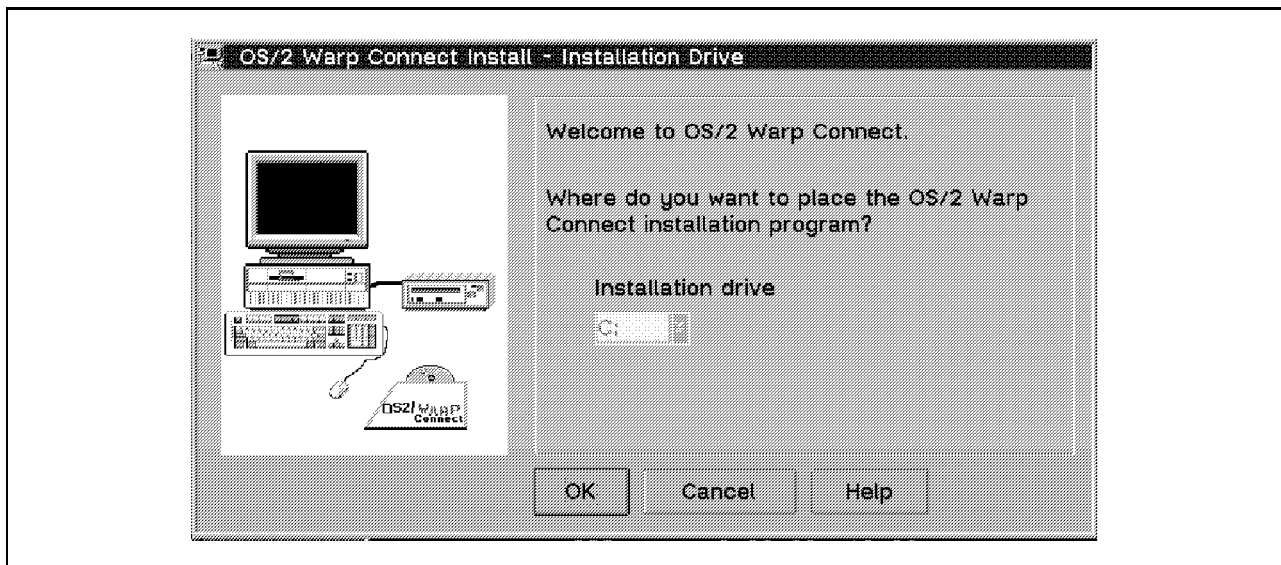


Figure 50. OS/2 Warp Connect Install - Installation Drive Window

3. The system has to copy some files for the installation. Then a new window appears to inform you that all necessary files have been copied (Figure 51 on page 27). Click on **OK**.

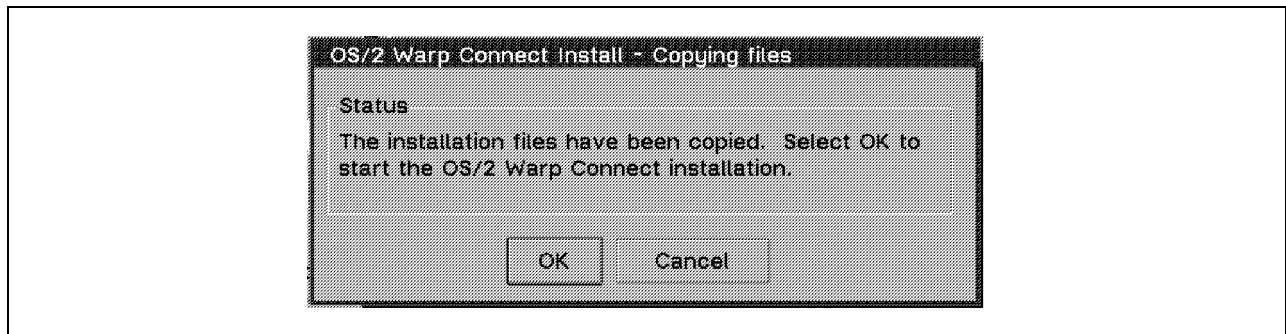


Figure 51. OS/2 Warp Connect Install - Copying files Window

4. On the Local versus Remote window (Figure 52) select **On this workstation** to specify local installation of the product and click on **OK**.

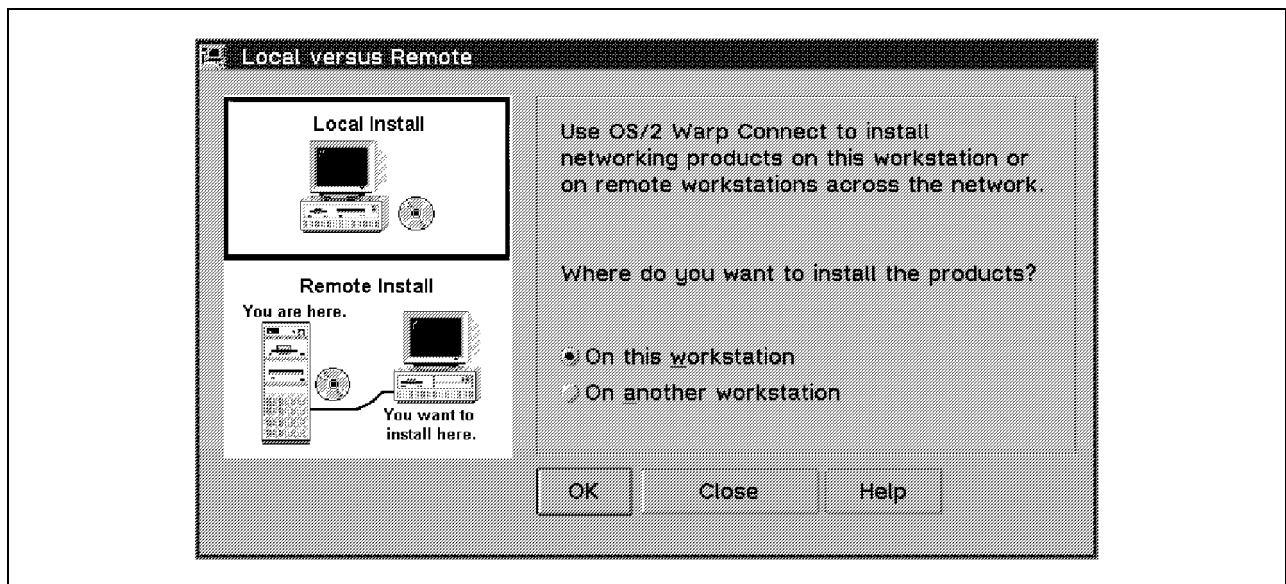


Figure 52. Local versus Remote Window

5. On the Installing OS/2 Warp Connect window (Figure 53 on page 28) select **Advanced Installation** so that you may choose LAN Requester instead of Peer for OS/2. The easy installation uses Peer for OS/2 as the client product. Click on **OK**.

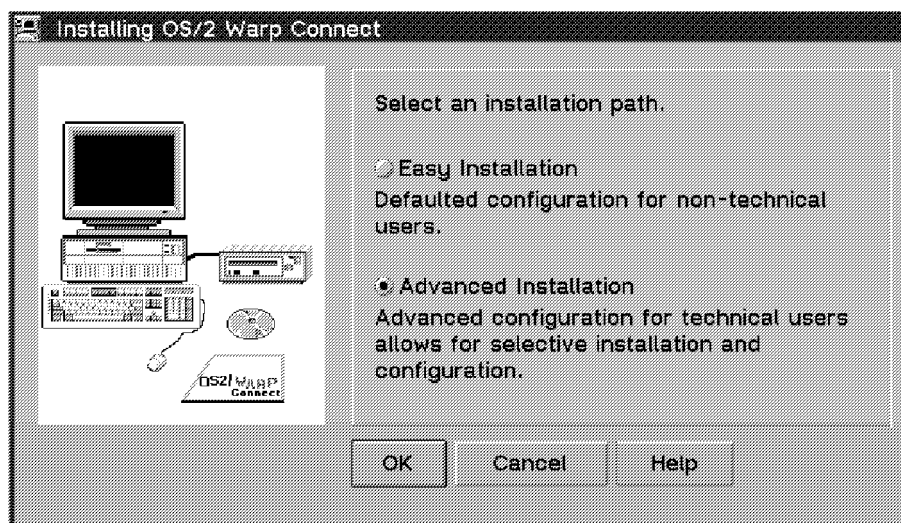


Figure 53. Installing OS/2 Warp Connect Window

6. On the Product Selection window (Figure 54) check **IBM client products** and select **IBM LAN Requester 4.0**. Then click on **OK**.

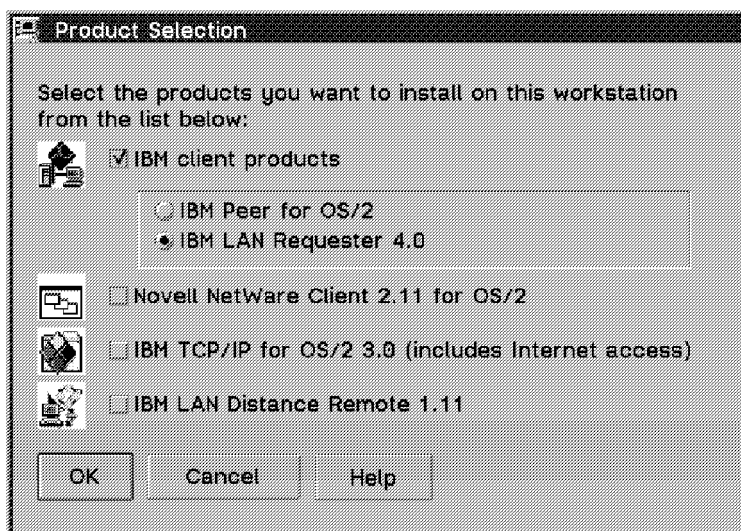


Figure 54. Product Selection Window

7. In the Set up selected products window (Figure 55 on page 29), the system shows a notebook where you have to qualify the network adapter and LAN Requester. Select the appropriate network adapter you are using.

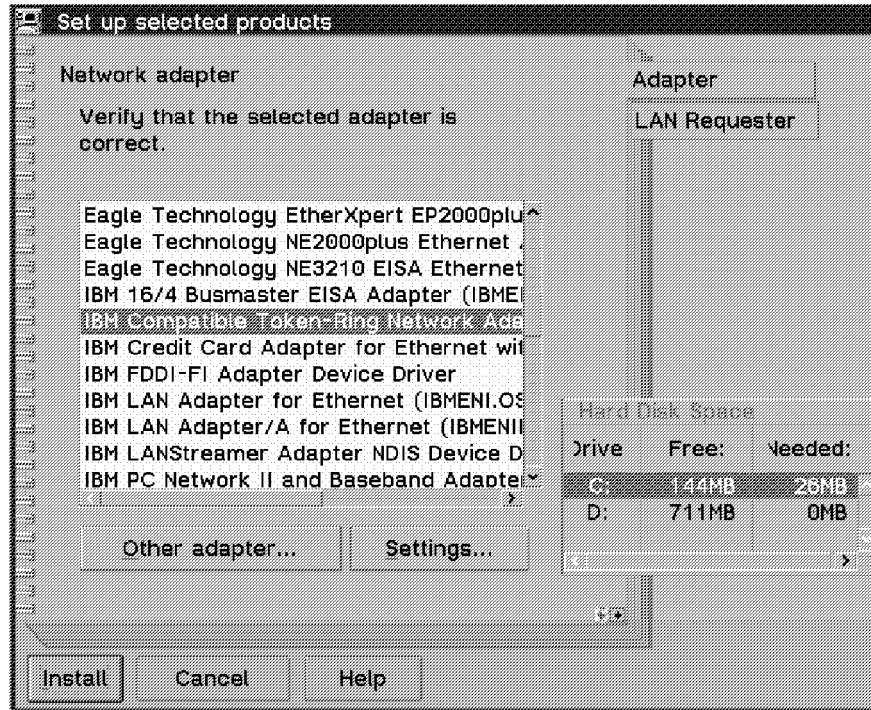


Figure 55. Set Up Selected Products Window—Adapter Options

Scroll to the next page (Figure 56 on page 30).

8. Select the drive on which you want to install IBM LAN Requester. Type in the requester name and domain name and select **NetBIOS** as the protocol. You cannot select NetBIOS over TCP/IP, because the server supports NetBIOS only.

In the *Requester name* field, type a unique name that will be used to identify the workstation in the OS/2 LAN Server domain. The requester name should be different from the user name so that the requester can receive messages from other users or applications in the LAN.

In the *Domain name* field, type the name of the OS/2 Server domain.

Then click on **Install**.

Network Messaging

The OS/2 LAN Server product provides a network messaging function for sending messages to and receiving messages from other users on the network. You must be logged on to be able to send and receive messages using network messaging. The network messaging function uses the Messenger service to accomplish these tasks. When a new message arrives on your workstation, the Messenger service adds it to the list of messages waiting to be read and, if you have configured the pop-up option, informs you of its arrival through a pop-up window. All messages you receive are saved in a message log on your workstation until you delete them.

Messages can be sent to users who are logged on or to workstations that are started and running the Messenger service. If you defined users and groups on the domain, they are listed as possible recipients of your messages in the List Users and List Groups windows when you are sending a new message.

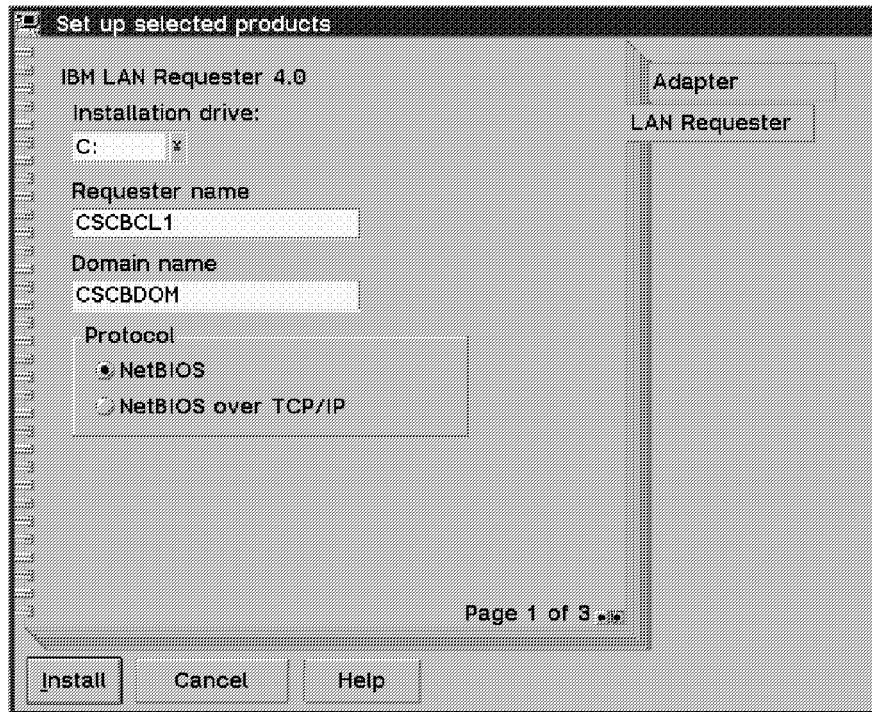


Figure 56. Set Up Selected Products Window—LAN Requester

Installing Other Clients

If you install clients in addition to the OS/2 Client, you must be sure to unify the requester name for each client.

9. On the window labeled
10. The setup is complete (Figure 57 on page 31) click on **Install** to start the installation.



Figure 57. The Setup Is Complete Window

11. The Desktop window (Figure 58) appears and requests that you shut down the workstation. Click on **OK**, shut down your machine, wait for the Shutdown completion message, and press the Ctrl, Alt, and Delete keys simultaneously to restart the system.

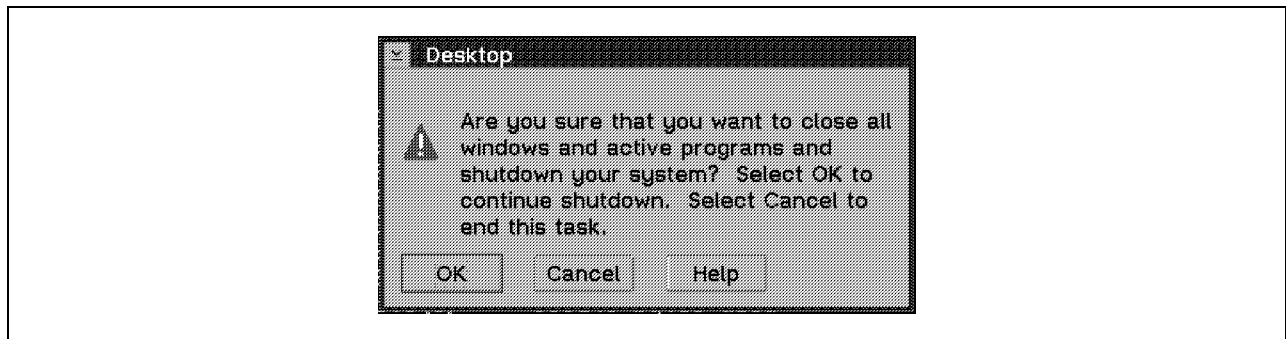


Figure 58. Desktop Window

12. After rebooting, the installation of LAN Requester begins. When the installation is complete, you will be informed that MPTS and IBM LAN Requester are installed. Click on **OK** to restart the workstation.

IBM LAN Requester will start automatically whenever the workstation is booted.

1.2.1 Installing DB2 for OS/2 - Single-User

To provide a development environment where developers can access DB2 servers from their own workstations, we recommend installing DB2 for OS/2 - Single-User. This product includes the functionality of the Administrator Toolkit, Client Application Enabler, and the DB2 Software Developer's Kit (SDK).

The Client Application Enabler provides run-time support for COBOL and other programming languages to enable applications to access local or remote database servers. It supports most communication protocols. The Client Application Enabler also provides a command line processor.

SDK provides the tools and environment you need to develop OS/2 applications that access DB2 servers and application servers that implement the Distributed Relational Database Architecture (DRDA). SDK provides COBOL precompilers.

To install DB2 for OS/2 - Single-User, go through the following steps:

1. Insert the DB2 for OS/2 - Single-User CD-ROM into the CD drive. At an OS/2 command prompt, type:

d:\INSTALL

where d is the drive letter of the CD-ROM, and press Enter.

2. On the Install window (Figure 59) check **Update CONFIG.SYS** and click on **OK**.

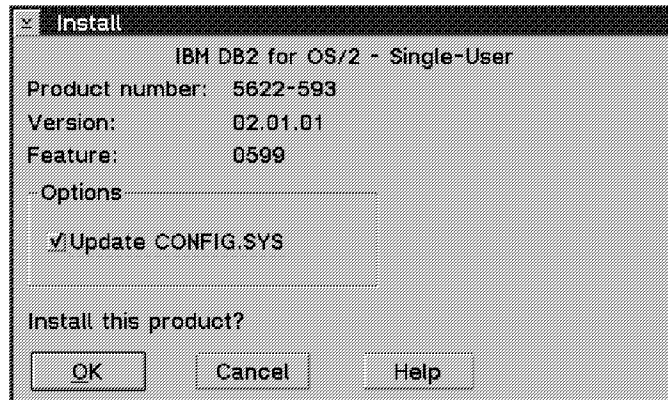


Figure 59. Install Window (IBM DB2 for OS/2 - Single-User)

3. On the Install - directories window (Figure 60) select:

- **Single-User**
- **Database Director**
- **Visualizer Flight**
- **Documentation**

Specify the file directory and click on **Install**.

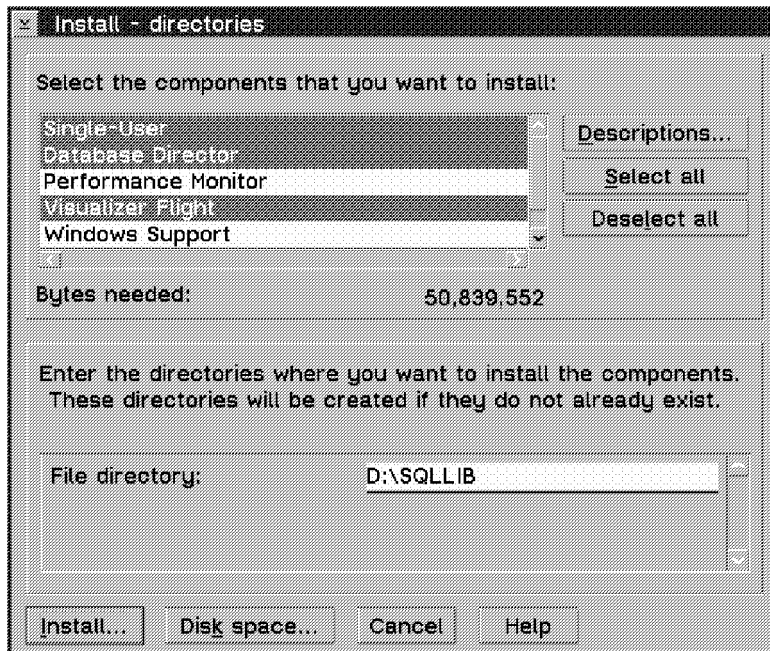


Figure 60. Install - directories Window (DB2 Single-User)

4. On the Installation and Maintenance window (Figure 61 on page 33) click on **OK**.

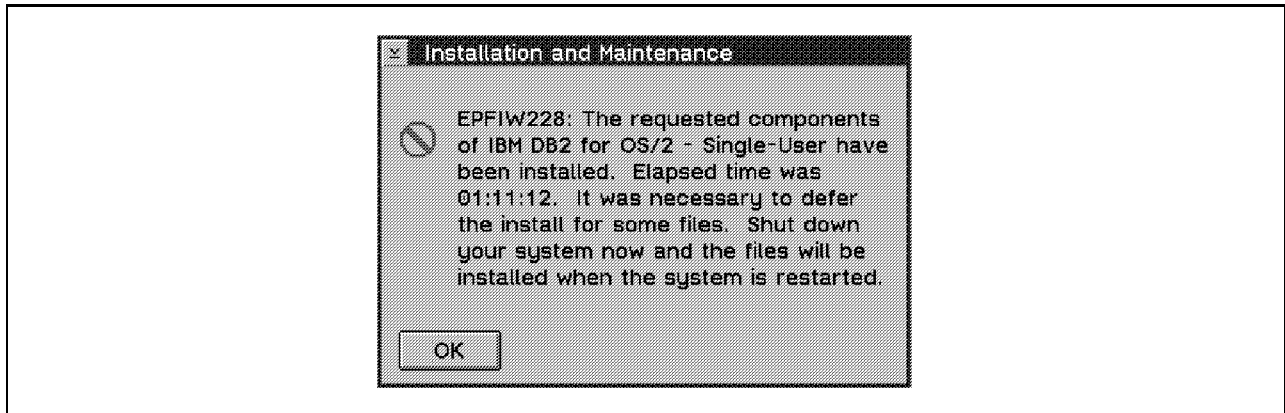


Figure 61. Installation and Maintenance Window (DB2 Single-User)

5. The Using DB2 for the first time window (Figure 62) appears. Click on **Logon dialog...**



Figure 62. Using DB2 for the First Time Window (DB2 Single-User)

6. When the Local Logon window (Figure 63 on page 34) appears, type `USERID` in the *User ID* field and the corresponding password `PASSWORD` in the *Password* field. Click on **OK**.

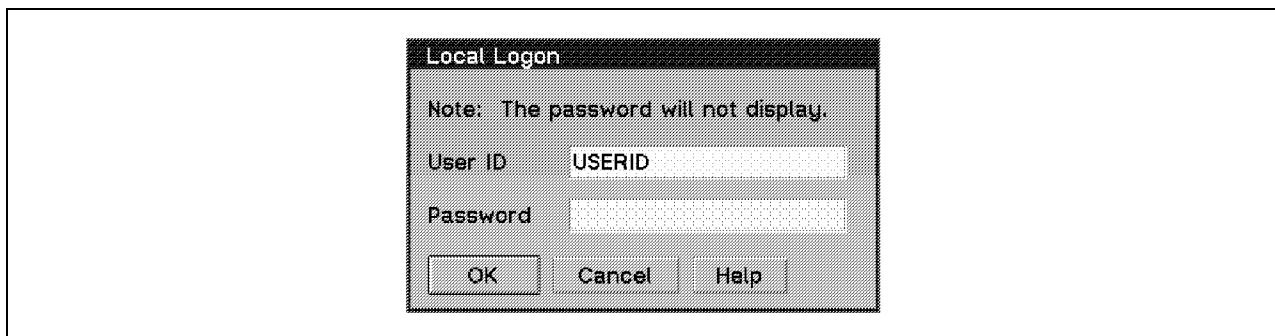


Figure 63. Local Logon Window (DB2 Single-User)

7. Click on **Create the sample database...** on the window Using DB2 for the first time.
8. On the Target drive selection window (Figure 64) take the default drive for the sample database and click on **OK**.

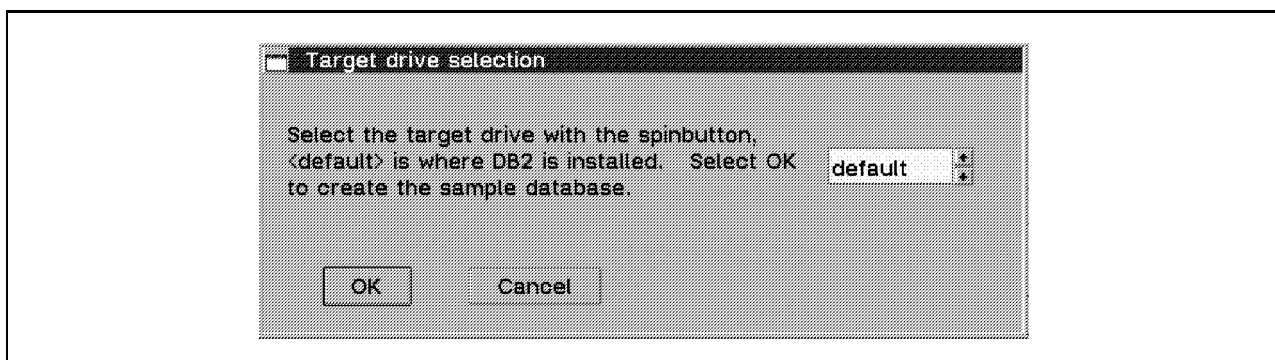


Figure 64. Target Drive Selection Window (DB2 Single-User)

9. On the Sample Database Creation Status window (Figure 65) click on **OK** and wait.

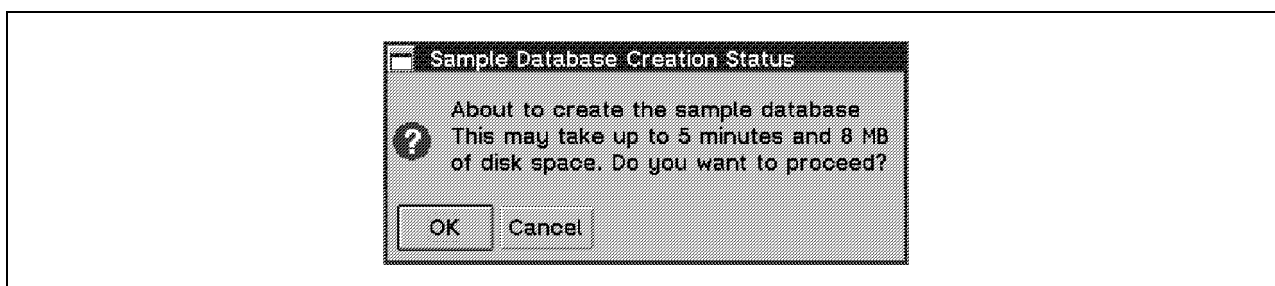


Figure 65. Sample Database Creation Status Window: About to Create (DB2 Single-User)

10. The window Sample Database Creation Status (Figure 66 on page 35) appears. Click on **OK**.

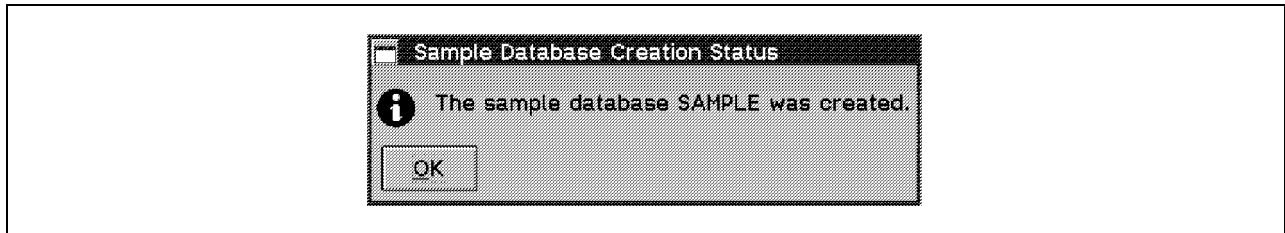


Figure 66. Sample Database Creation Status Window: Creation Completed (DB2 Single-User)

11. Edit the CONFIG.SYS file and change the COBCPY environment variable to
`SET COBCPY=d:\SQLLIB\INCLUDE\COBOL_A`
12. Shut down and reboot your workstation.

The installation of DB2 for OS/2 - Single-User is complete.

1.2.2 Installation of Software Developer's Kit for OS/2

The Software Developer's Kit for OS/2 enables you to develop your applications on your own OS/2 workstation that accesses DB2 server databases. You need only install Software Developer's Kit for OS/2 on a workstation where DB2 for OS/2 - Single-User is not installed.

To install SDK, do the following:

1. Insert the IBM DATABASE 2 Software Developer's Kit for OS/2 CD-ROM into the appropriate drive. At an OS/2 command prompt, type:
`d:\INSTALL`
 where d is the drive letter of the CD-ROM, and press Enter.
2. The IBM DATABASE 2 Software Developer's Kit for OS/2 Installation window appears (Figure 67).

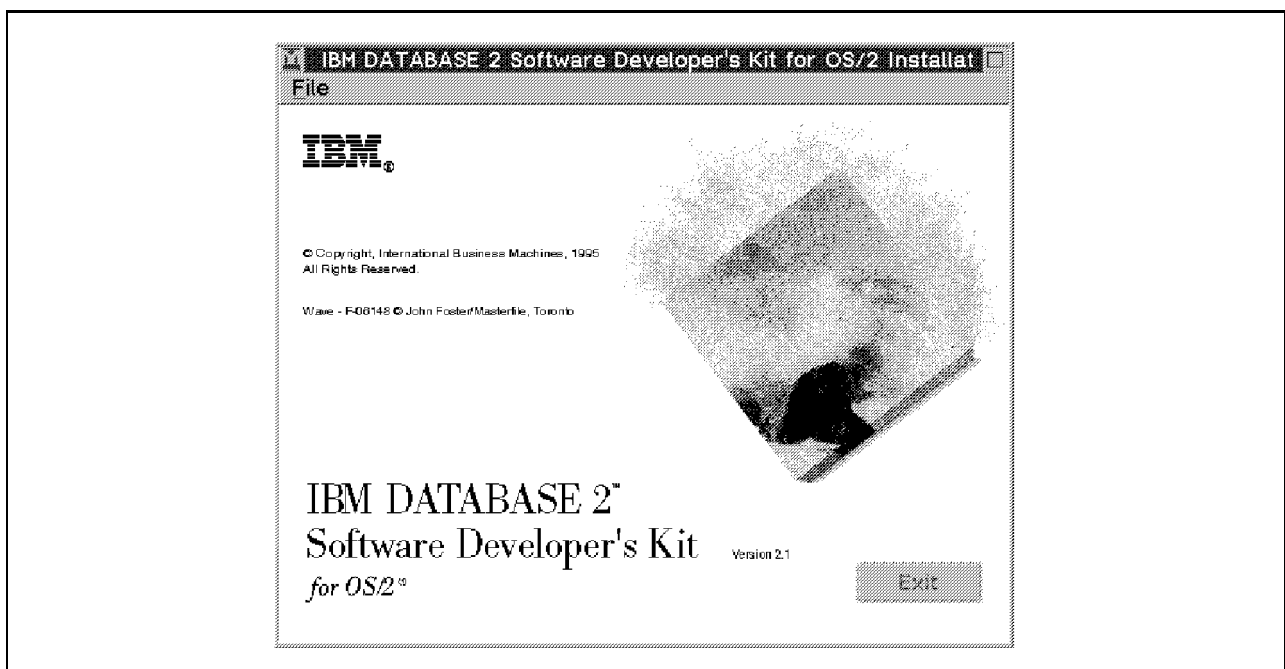


Figure 67. IBM DATABASE 2 Software Developer's Kit for OS/2 Installation Window

3. On the Install window (Figure 68 on page 36) ensure that **Update CONFIG.SYS** is checked. Click on **OK**.

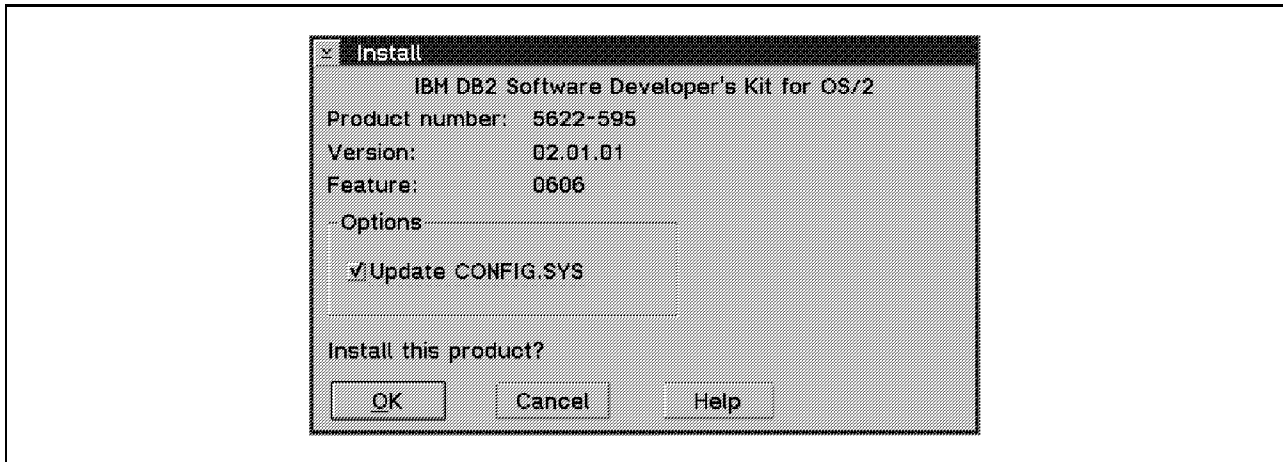


Figure 68. Install Window (IBM DB2 Software Developer's Kit for OS/2)

4. On the Install - directories window (Figure 69), you qualify the directories you are installing. Click on **Select all** to install the entire product. Specify the file directory name and click on **Install....**

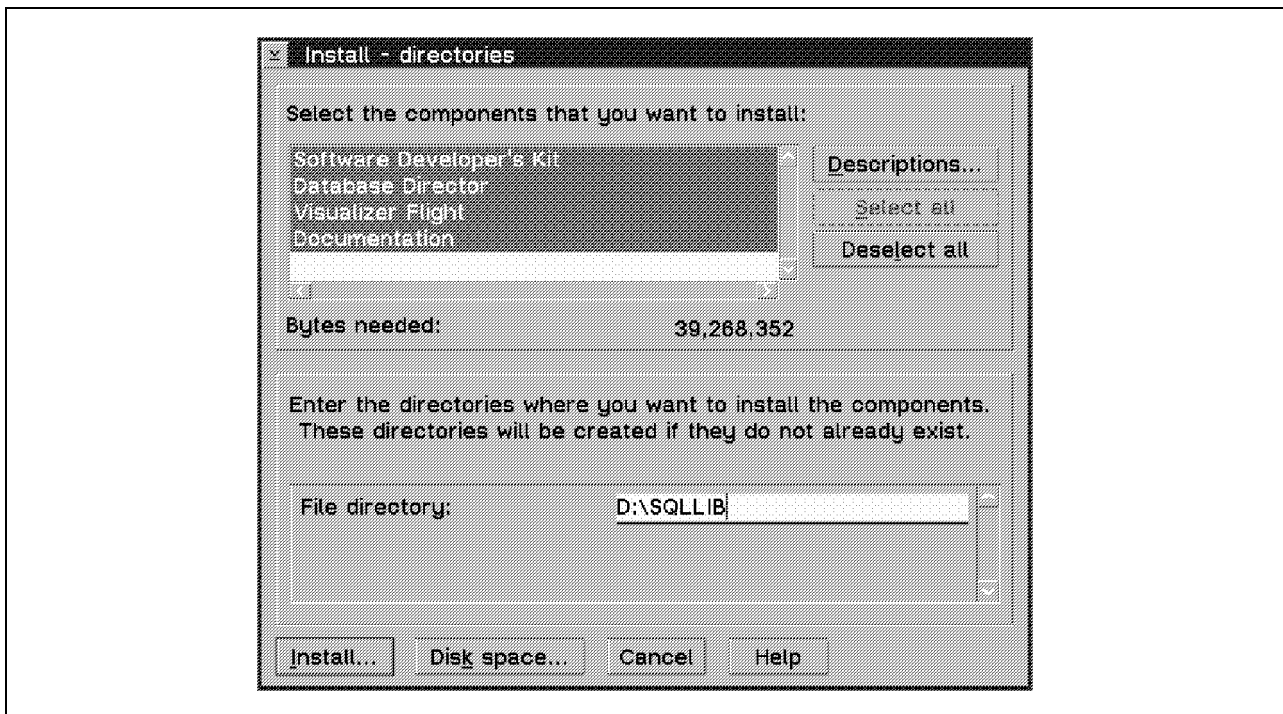


Figure 69. Install - Directories Window (DB2 SDK)

5. Use the Install - progress window (Figure 70 on page 37) to monitor the progress of the installation.

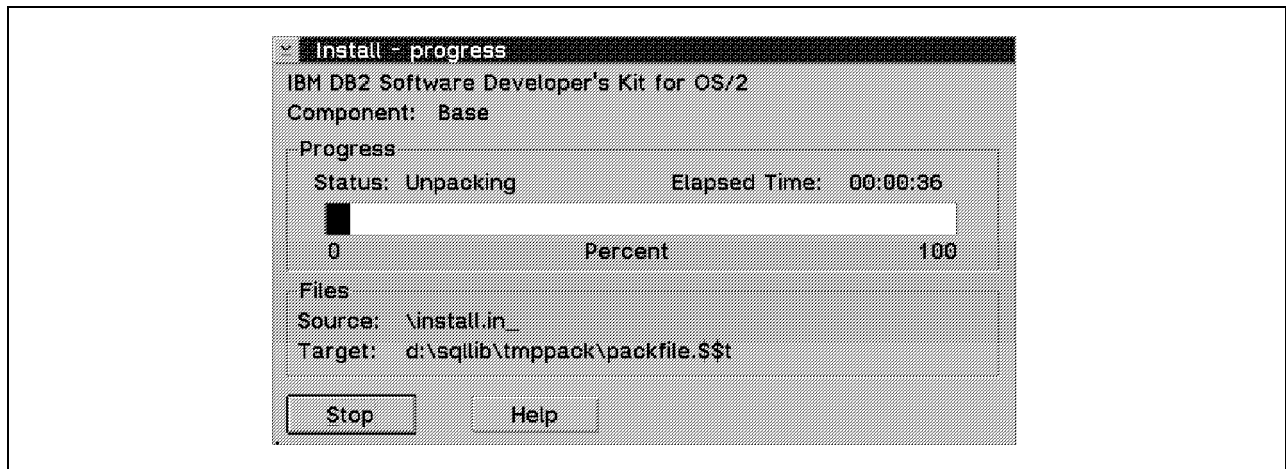


Figure 70. Install - Progress Window (DB2 SDK)

6. The Installation and Maintenance window informs you of the successful installation (Figure 71). Click on **OK** and shut down your workstation.

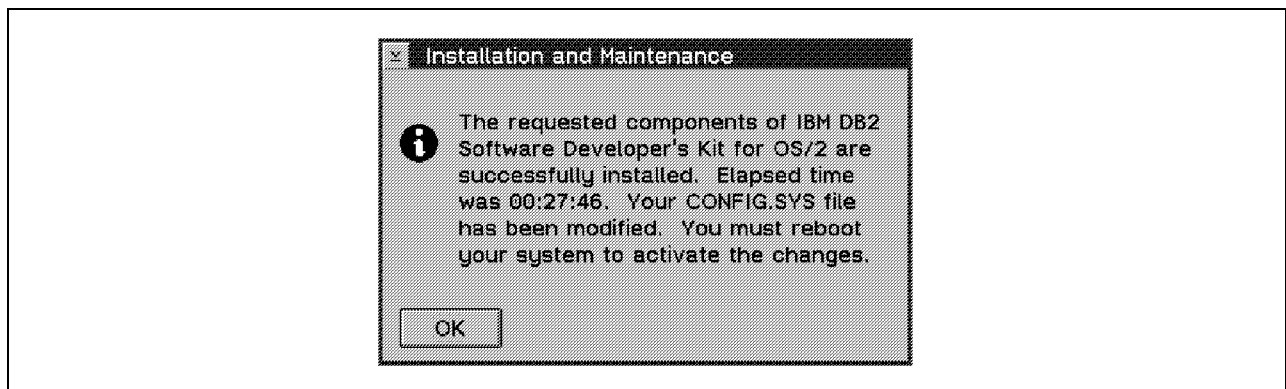


Figure 71. Installation and Maintenance Window (DB2 SDK)

7. To configure the DB2 for OS/2 on this client machine, you must follow the steps in Chapter 1.1.4, "Configuring DB2 for OS/2 on the Server" on page 18. The workstation name you specify in this configuration must be unique in the entire environment. After this configuration, your client is able to access data from the server.
8. Edit the CONFIG.SYS file and change the COBCPY environment variable to

```
SET COBCPY=d:\SQLLIB\INCLUDE\COBOL_A
```
9. Shut down and reboot your workstation.

Your workstation is now ready for application development. To extend your environment and access database, install IBM VisualAge for COBOL for OS/2 on the workstation.

1.2.3 DB2 Client Setup

The DB2 Client Application Enabler for OS/2 includes a new graphical user interface (GUI) tool called the *DB2 Client Setup*. This tool enables the client to catalog nodes and databases, test connections, and bind packages to databases and it performs traces for problem determination.

To configure the DB2 Client Setup, complete the following steps:

1. Define the NetBIOS node name for the workstation. To do this, use the same procedure as described in 1.1.4, "Configuring DB2 for OS/2 on the Server" on page 18 up to the point where you reach the NetBios system configuration change in the file PROTOCOL.INI. But do not use the same node name as before, because it has to be unique. You need not change the PROTOCOL.INI.

2. You can now configure the DB2 Client Setup as follows:

Double-click on the **IBM DATABASE 2** icon on the Desktop. The IBM DATABASE 2—Icon View window appears. Double-click on the **Client Setup** icon. The window DB2 Client Setup - Product Information appears (Figure 72). Click on **OK** to start the client configuration.

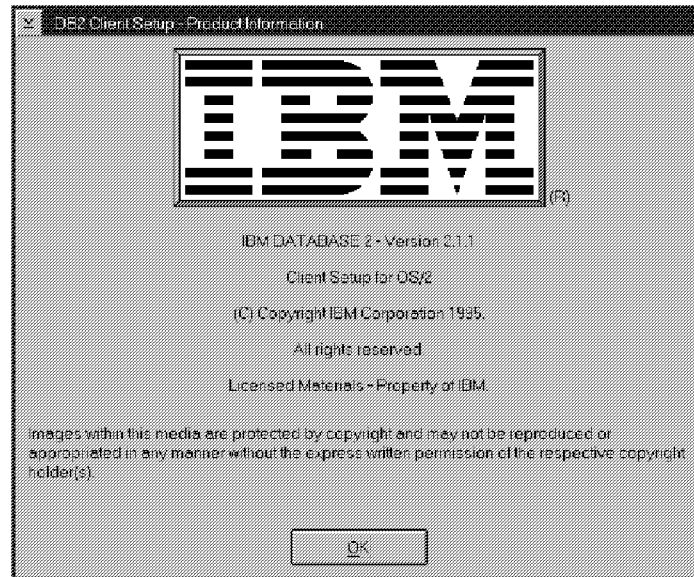


Figure 72. DB2 Client Setup - Product Information Window

3. On the DB2 Client Setup window (Figure 73) select **Node** from the menu bar and **New...** from the pull-down menu to create a new node.

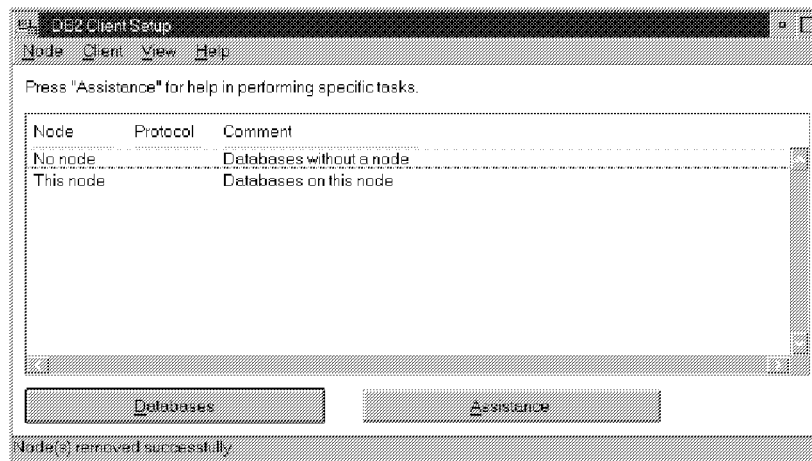


Figure 73. DB2 Client Setup Window (before adding node)

4. In the New Node window (Figure 74 on page 39), type in the (unique) name of the new DB2 Client node, and (optionally) a comment. Select **NetBIOS** as the protocol. In the *Workstation Name* field, enter the name of the DB2 Server. In the *Adapter* field, enter 0. Click on **OK**.

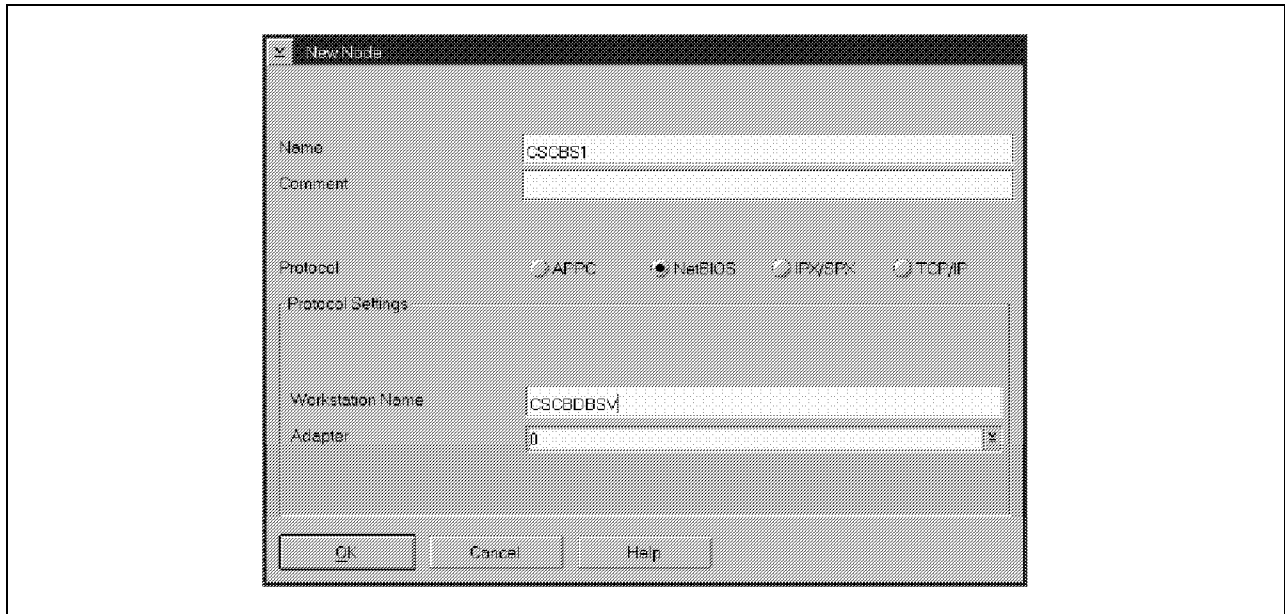


Figure 74. New Node Window

5. The DB2 Client Setup window appears again with the new information about the node (Figure 75).

Before a client application can access a remote database, the database must be cataloged on the client node. The database manager uses the information in the database directory along with the information in the node directory to establish a connection to the database.

To catalog a database by using a DB2 Client Setup, double-click on the node you have just created.

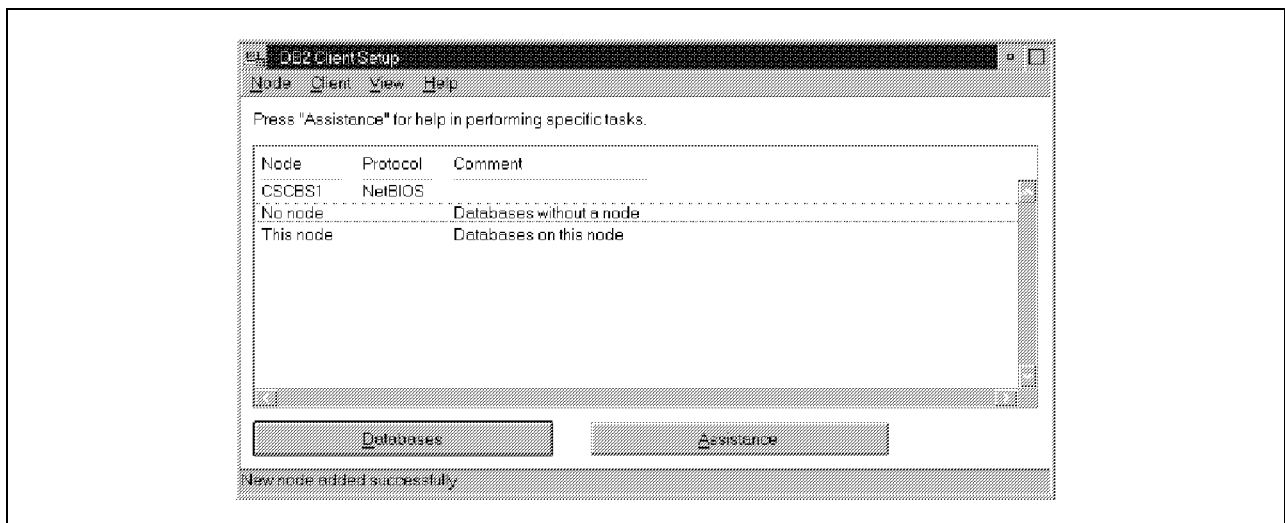


Figure 75. DB2 Client Setup Window (after adding node)

6. The DB2 Client Setup - Databases window is displayed (Figure 76 on page 40). Select **Database** from the menu bar and **New...** from the pull-down menu.

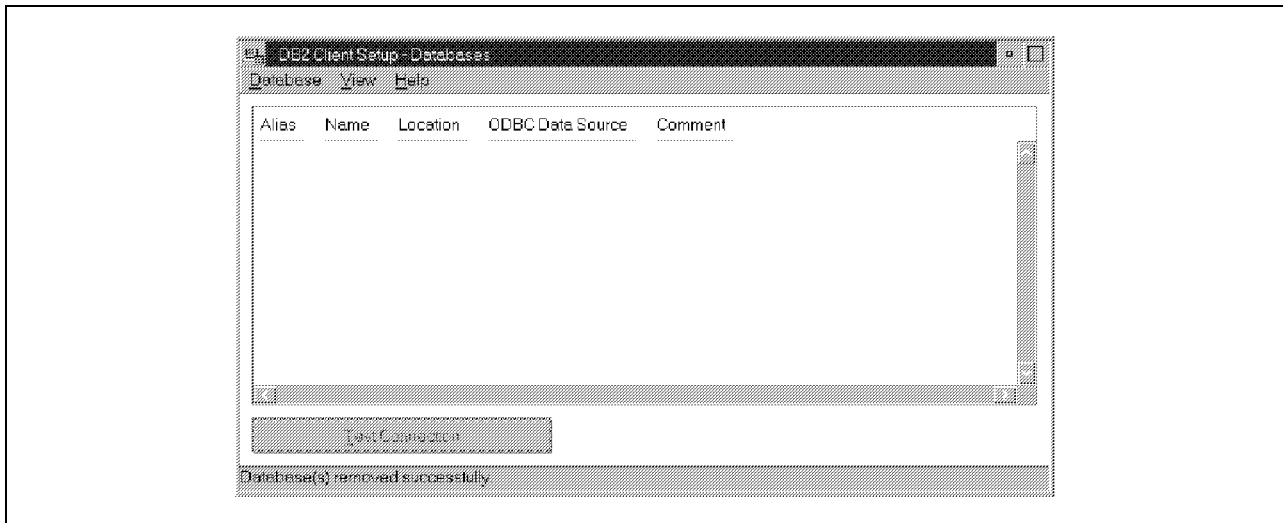


Figure 76. DB2 Client Setup - Databases Window

7. In the New Database window (Figure 77 on page 41), to connect to the Sample database, type in `SAMPLE` for the name of the database and `SAMPLE` for the alias. The database name must be the real name of the database. The alias name can be any name you want. You must use this alias as the database name in your source of the application that accesses the database.

Because you want to connect to the database of the server, select **Other Node** and in the *At Node* field enter the node name.

Click on **Test Database Connection** to run the connection.

Note

The window in Figure 77 on page 41 belongs to the DB2 Client Setup on a machine that has installed DB2 for OS/2 - Single-User. When you set up the DB2 Client on a machine that has Software Developer 's Kit for OS/2 installed instead of DB2 for OS/2 - Single-User, the New Database window looks different. It gives you no radio buttons to select one node or another; you can only locate to another node.

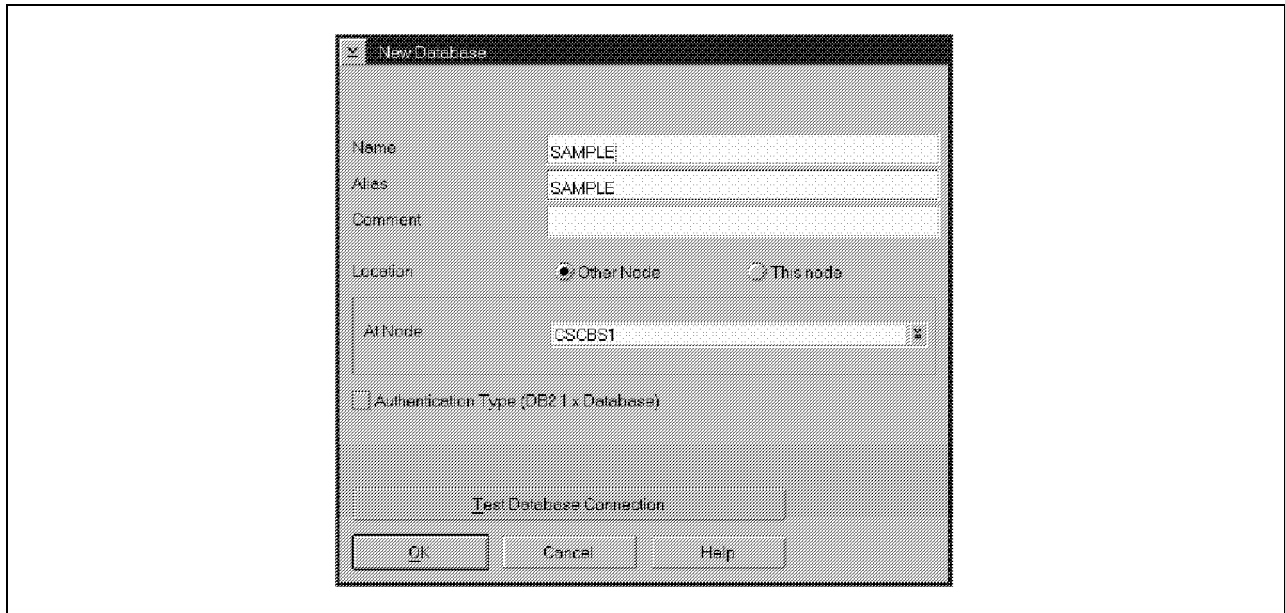


Figure 77. New Database Window

8. In the Client Application Enabler/2 window (Figure 78), enter the required user ID and password. Select **Share** to share the server database with other users and click on **OK**.

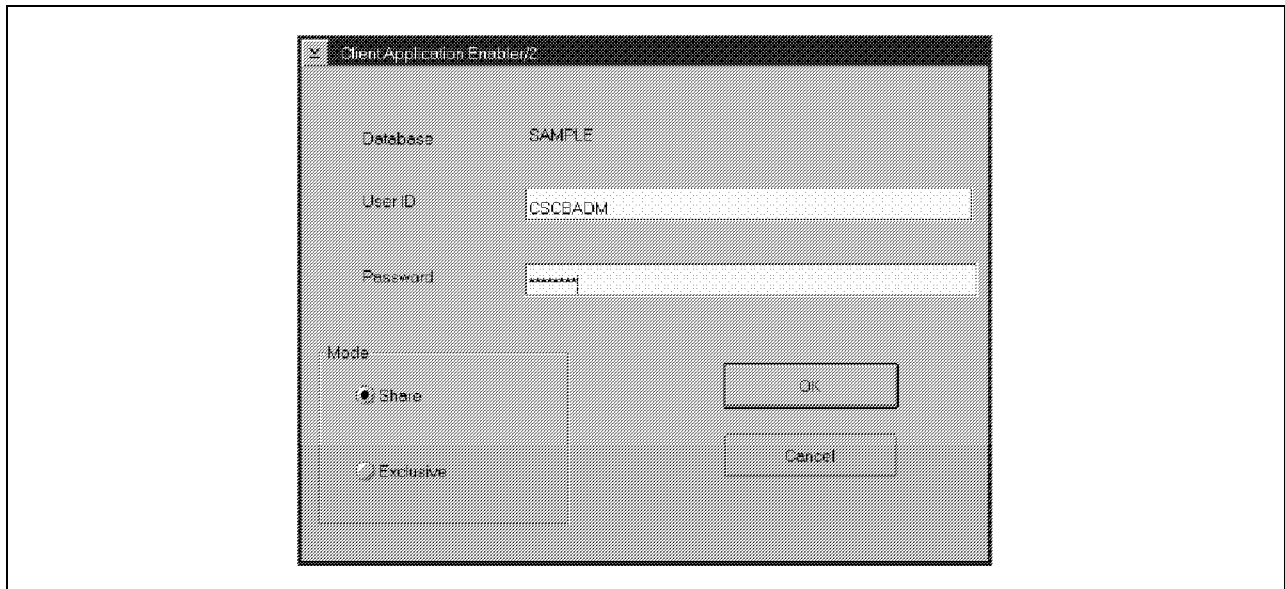


Figure 78. Client Application Enabler/2 Window

9. The New Database window appears again. Click on **OK** in this window and close all windows that are still open.
10. If you want to know which database directories exist, open the Command Line Processor by double-clicking on the **Command Line Processor** icon in the IBM DATABASE 2 - Icon View window (Figure 79 on page 42).

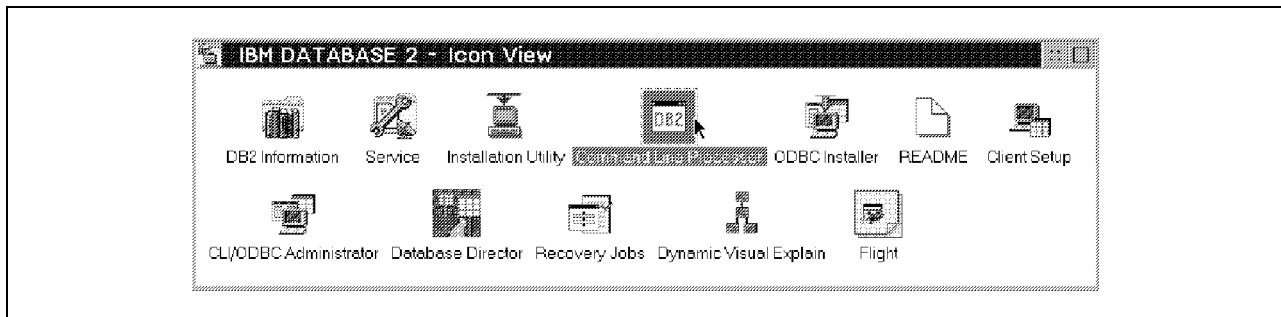


Figure 79. IBM DATABASE 2 - Icon View Window (DB2 Client)

11. On the Command Processor window (Figure 80), type:

`list database directory`

at the command prompt.

A list of all database directories appears.

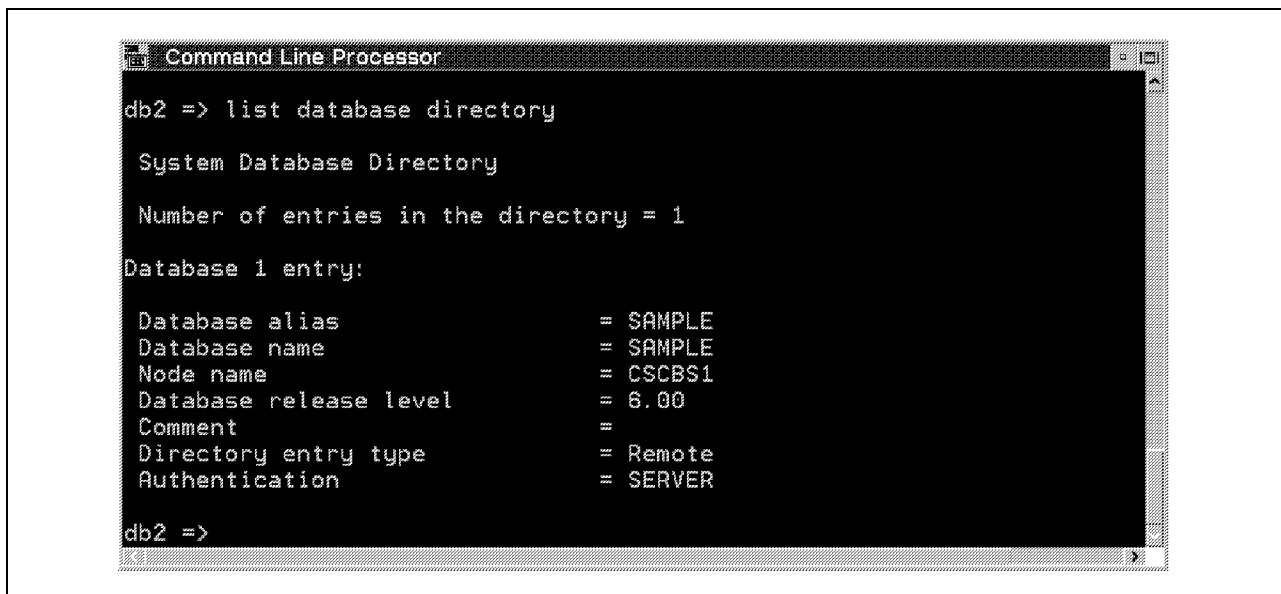


Figure 80. Command Line Processor Window

DB2COMM

The DB2COMM environment variable on the server determines which protocols will be enabled when the database manager is started. Use this command in an OS/2 window to check your configuration. For example, if your DB2 server is configured to start NetBIOS, the environment variable would be DB2COMM=NetBIOS.

CATALOG and CONNECT TO DATABASE

Cataloging a database adds a database entry to the database directory. Information recorded in this directory includes the local name (alias) given to a remote database and the node on which the database is located.

A database can be cataloged before cataloging the node on which it resides. However, a warning that the node has not yet been cataloged will be displayed.

The CATALOG DATABASE command is also used on client nodes to catalog databases that reside on database server machines. At the client node, you can catalog the database by using either the DB2 Client Setup or the CATALOG DATABASE command.

To use the command, type:

```
catalog database name as alias at node db2node authentication type
```

- *name* is the real name of the remote database.
- *alias* is the alias you want to use for this database. If you do not provide an alias, the default is the same as the database name. The real database name and the alias are stored in the client's database directory. To reference the cataloged database, you must use the alias.
- *db2node* is the value you used for *db2node* when you cataloged the node that represents the instance where the database resides.
- *type* is the authentication type used with this database. This parameter is optional.

At the client, you can also connect to the server database using the command:

```
CONNECT TO server-name { USER authorization-name USING password }
```

- *server-name* identifies the application server. This server name is a database alias that identifies the application server.
- *authorization-name* identifies the ID of the user trying to connect to the application server.
- *password* identifies the password of the user ID trying to connect to the application server.

The portion of the command in parentheses is not necessary. If you leave it out and are not yet logged on, the logon screen of UPM prompts you to log on.

DB2 for MVS supports a 16-byte location name while DB2 Version 2 supports the use of only an 8-byte database-alias name on the SQL CONNECT statement. However, the database-alias name can be mapped to an 18-byte database name through the Database Connection Service Directory.

When the CONNECT TO statement is executed, the application process must be in the connectable state.

1.2.4 Installing IBM VisualAge for COBOL for OS/2

To install IBM VisualAge for COBOL for OS/2 Single-User, follow these steps:

1. Insert the IBM VisualAge for COBOL for OS/2 CD-ROM into the CD drive. At the OS/2 command prompt, type:

```
d:\INSTALL
```

where d is the drive letter of the CD-ROM, and press Enter.

2. The Installation window appears (Figure 81). If necessary, read the displayed instructions and click on **Continue**.

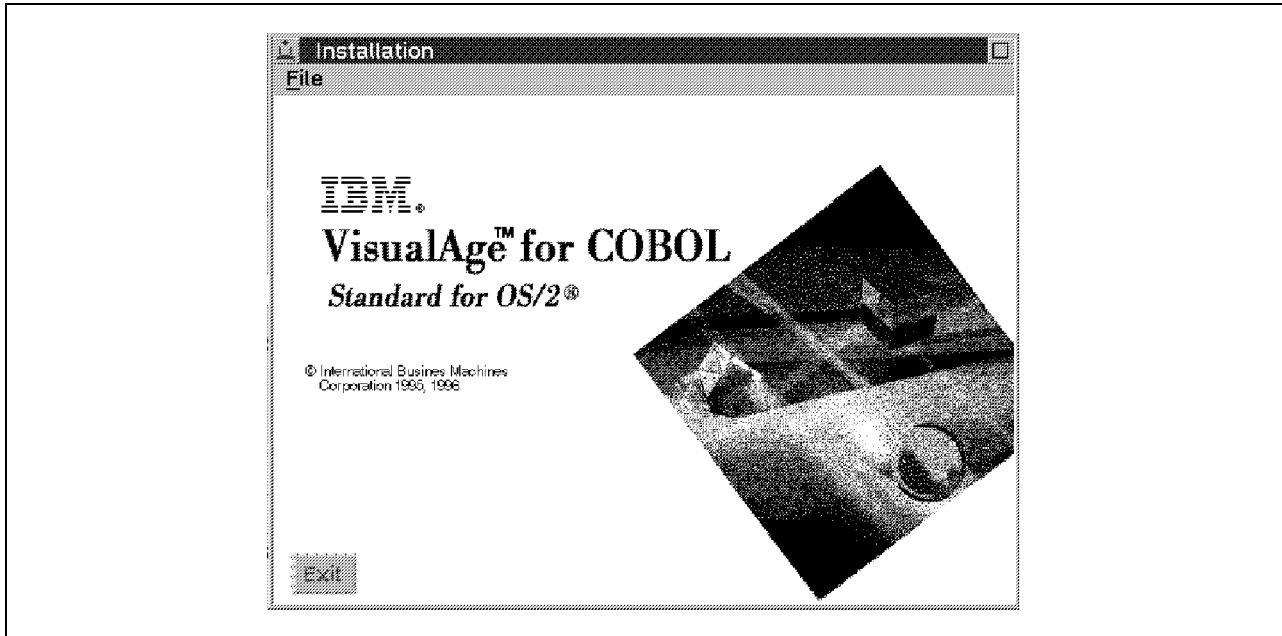


Figure 81. Installation Window (IBM VisualAge for COBOL, Standard for OS/2)

3. The Install window then appears (Figure 82). The default selection to update the CONFIG.SYS indicates that the installation program will automatically make changes to the CONFIG.SYS file. Click on **OK** to install the product.

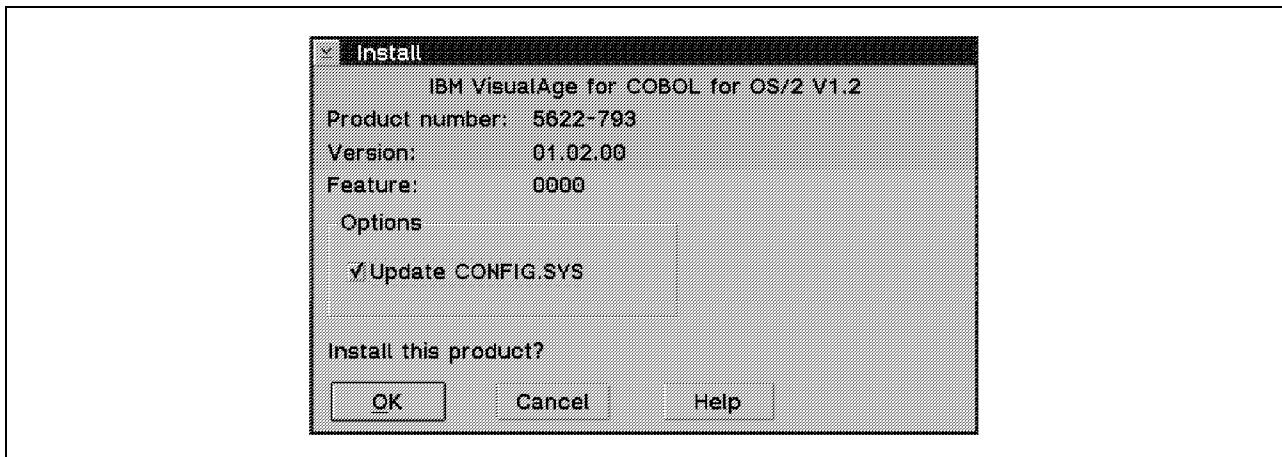


Figure 82. Install Window (VisualAge for COBOL for OS/2)

4. On the Install - directories window (Figure 83 on page 45) accept all defaults and change the drive if necessary. Click on **Install....**

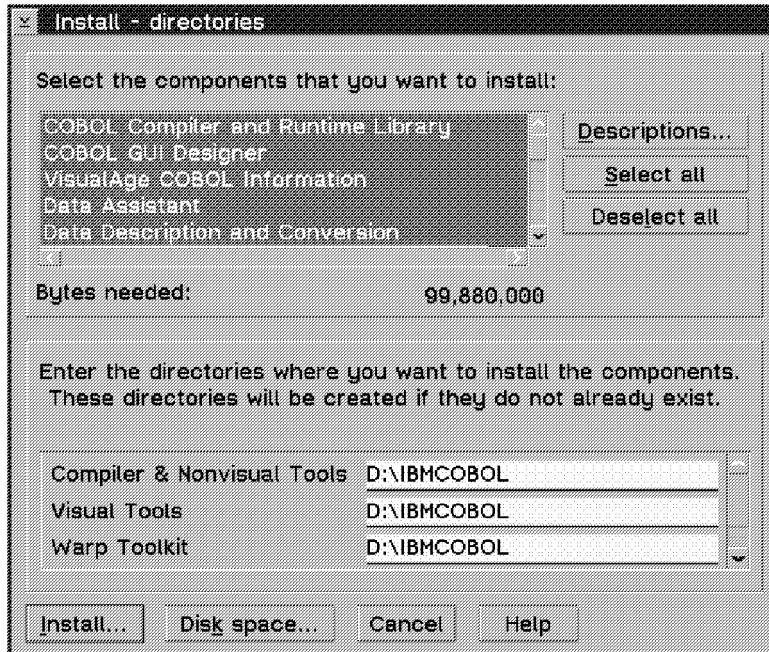


Figure 83. Install - directories Window (VisualAge for COBOL)

5. The Install - progress window (Figure 84) shows which files are being transferred and the status of the installation.

No other action is possible while the installation is in progress.

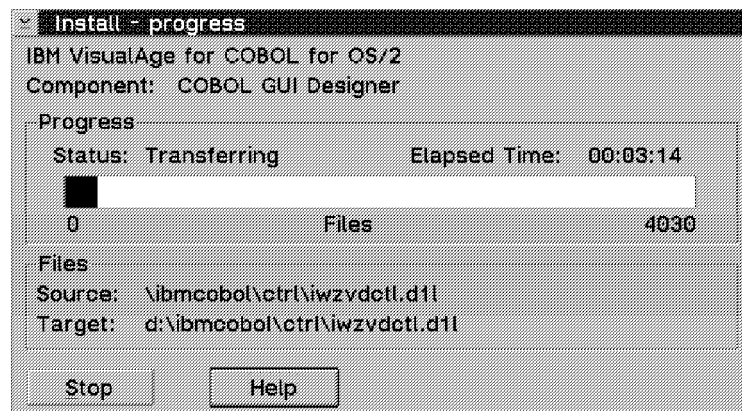


Figure 84. Install - Progress Window (VisualAge for COBOL)

6. Toward the end of the installation, the VisualAge for COBOL Installation window (Figure 85 on page 46) appears and informs you about the next steps. Click on **OK**.

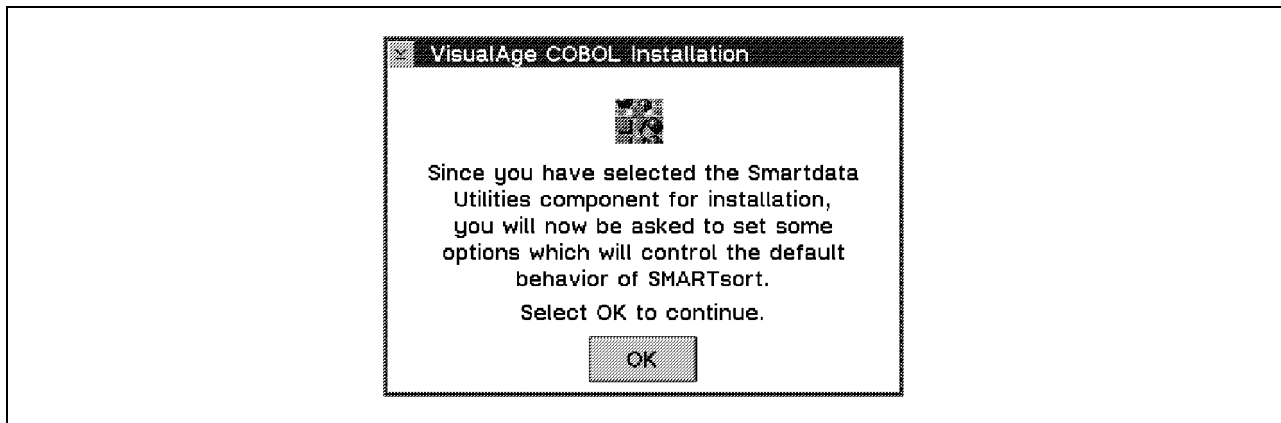


Figure 85. VisualAge COBOL Installation Window

7. On the SMARTsort 1.1.0 Defaults window (Figure 86), accept the default memory and the work directory. Click on **OK**.

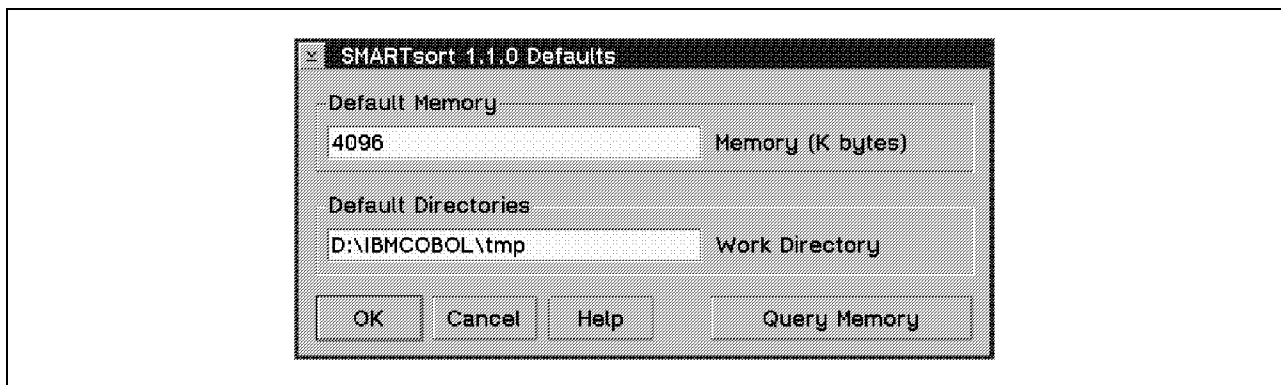


Figure 86. SMARTsort 1.1.0 Defaults Window

8. The VisualAge COBOL Install window (Figure 87 on page 47) asks whether you want to change the default compiler options during or after the installation. Click on **Cancel**.

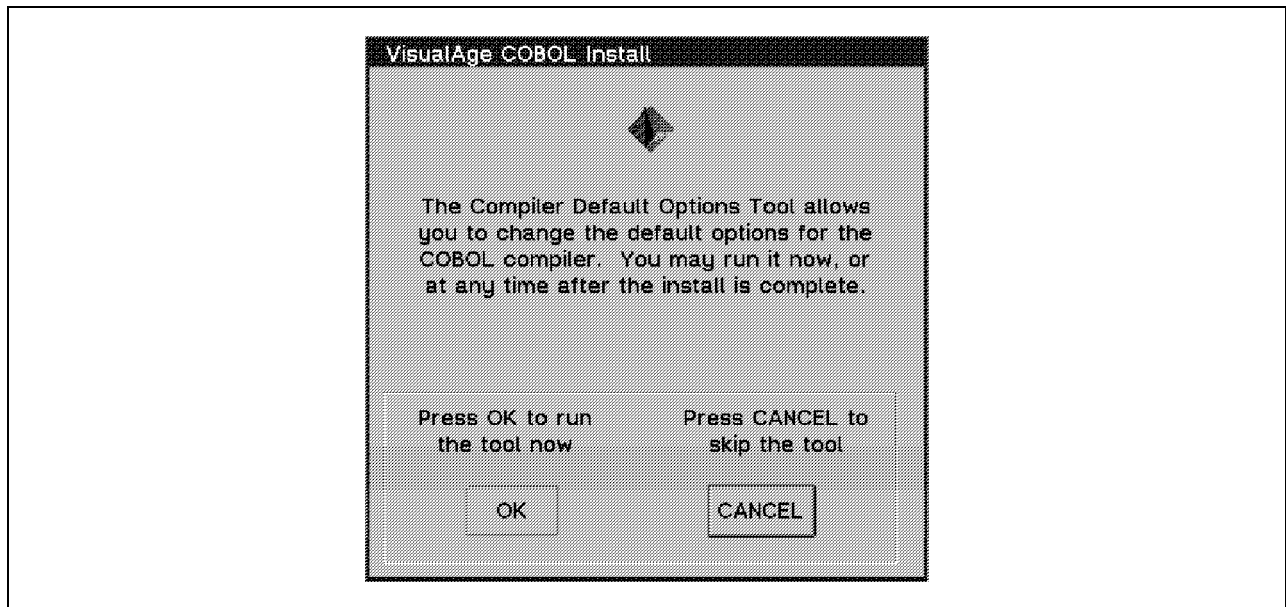


Figure 87. VisualAge COBOL Install Window

9. The Installation and Maintenance window (Figure 88) informs you of the successful installation and tells you to reboot the workstation. Click on **OK** and shut down the workstation.

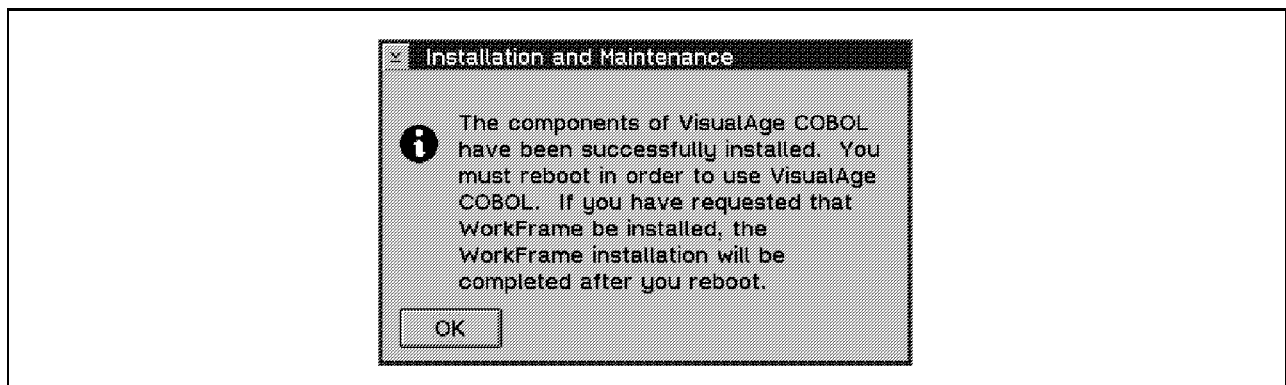


Figure 88. Installation and Maintenance Window (VisualAge for COBOL)

When the system has rebooted, WorkFrame installation is complete. Once WorkFrame is installed, the installation of IBM VisualAge for COBOL for OS/2 is finished. You can begin developing your application that accesses DB2 locally or remotely on the server.

1.2.5 Executing a COBOL Application That Accesses a Remote Database

After you have set up the DB2 for OS/2 client environment and cataloged the SAMPLE database, you can execute a COBOL application on the client machine to access the SAMPLE database from the remote server. This section describes the steps needed to execute a sample COBOL application, SALESDEP, in this OS/2 client/server environment. The source files for the SALESDEP application are included in the diskette that accompanies this book. Copy these files into your hard disk drive in one of the steps described below.

To execute SALESDEP by using the sources files, follow these steps:

1. Double-click on the **VisualAge COBOL** icon on the OS/2 Desktop. The VisualAge COBOL—Icon View window appears (Figure 89 on page 48).



Figure 89. VisualAge for COBOL—Icon View Window

2. Double-click on **Templates**. The window Templates - Icon View appears (Figure 90). Move the mouse pointer to the **COBOL Project** icon. Using mouse button 2, move the icon to the empty space on the Desktop. Release mouse button 2 and you will see the **COBOL Project** icon displayed on the Desktop.

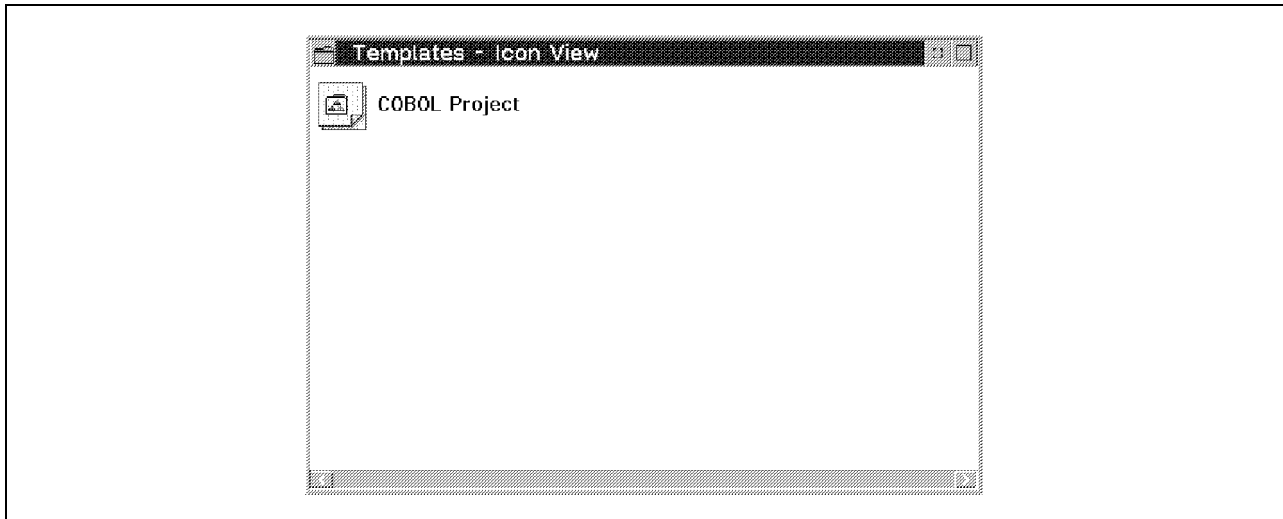


Figure 90. Templates - Icon View Window

3. Move your mouse pointer to the **COBOL Project** icon on the Desktop and click mouse button 2. A pop-up menu appears. Select **settings** from the pop-up menu. The COBOL Project - Settings window is displayed (Figure 91 on page 49).

Type `SALESDEP.EXE` in the *Name* field.

Type `SALESDEP.MAK` in the *Makefile* field.



Figure 91. COBOL Project - Settings Window: Target Page

4. Click on the **Location** tab. The Location page appears (Figure 92).

In the *Source directories for project files* field, type, type:

d:\IBMCOBOL\SALESDEP

and

d:\IBMCOBOL\SALESDEP\RT_OS2

(Type these paths in upper case.)

Here, d: is the drive where you installed IBM VisualAge for COBOL for OS/2.

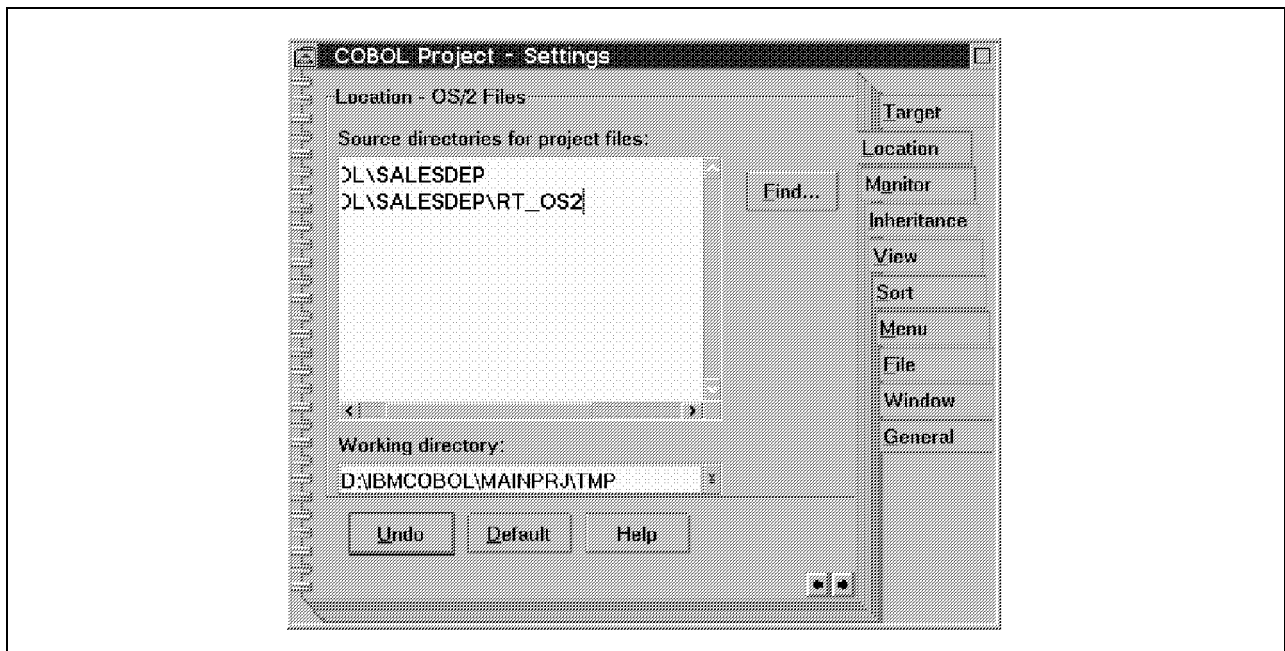


Figure 92. COBOL Project - Settings Window: Location Page (Source Directories)

Because you do not have the SALESDEP directories, the Create Directories window (Figure 93 on page 50) appears automatically when you click anywhere on the COBOL Project - Setting window or press the Tab key. You are asked in the Create Directories window whether you want the directories created. Click on **Yes** to create the two directories.

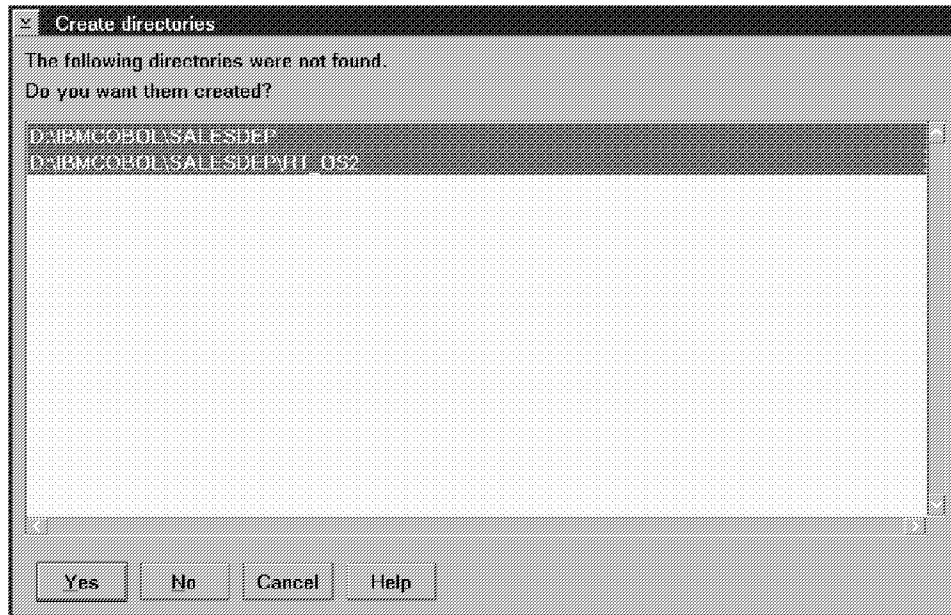


Figure 93. Create Directories Window

You return to the COBOL Project - Settings window. On the Location page (Figure 94), select `d:\IBMCOBOL\SALESDEP\RT_OS2` as the working directory.

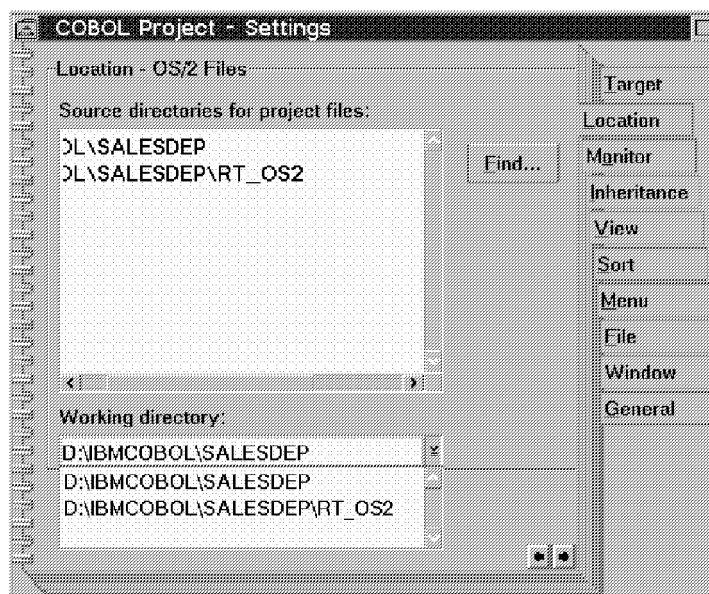


Figure 94. COBOL Project - Settings Window: Location Page (Working Directory)

- Click on the *Inheritance* tab. The Inheritance page appears. You can select the project type from which your project will inherit (Figure 95 on page 51).

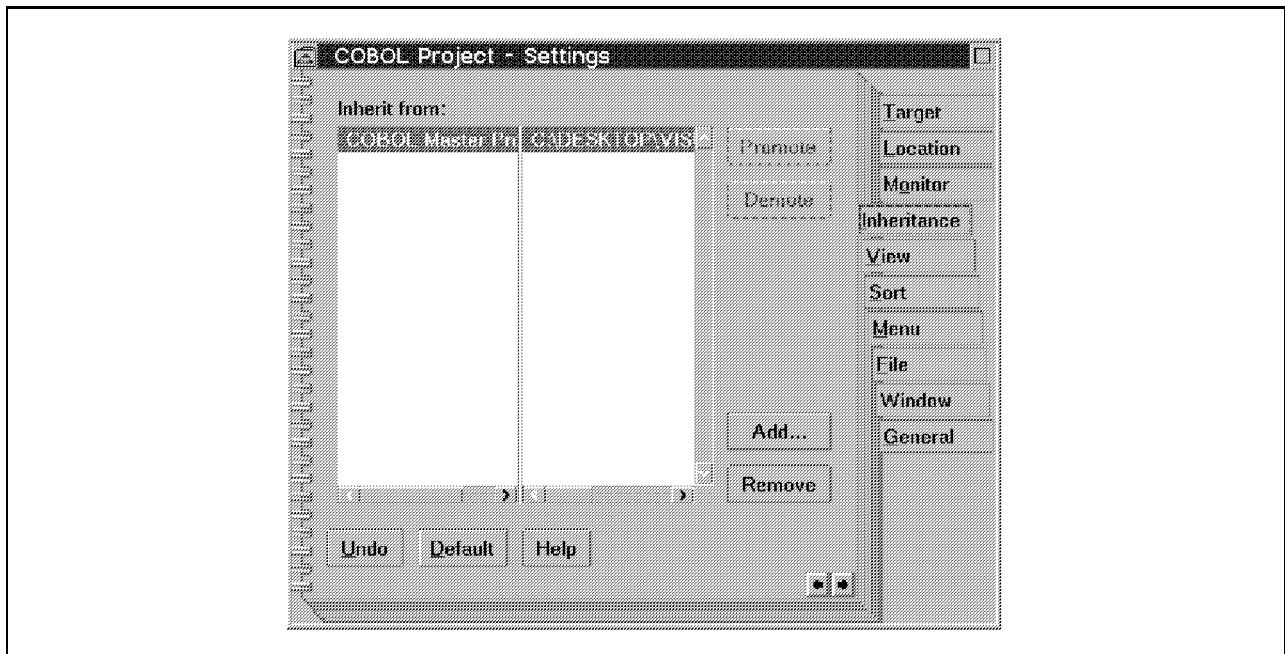


Figure 95. COBOL Project - Settings Window: Inheritance Page

Because SALESDEP is a GUI project, you have to change the default settings of the COBOL project. Click on **Remove**. and the default settings for the COBOL Master Project will be removed.

Click on **Add...**. The window Select a project to Inherit from appears. If your hard disk drive C is File Allocation Table (FAT) formatted, select **C:\DESKTOP\VISUALAG\WORKS\COBOL_G1** as the file SALESDEP inherits from (Figure 96). If your hard disk drive C is High Performance File System (HPFS) formatted, select **C:\Desktop\VisualAge COBOL\Works\COBOL GUI Designer Master Project**. Click on **Inherit**.

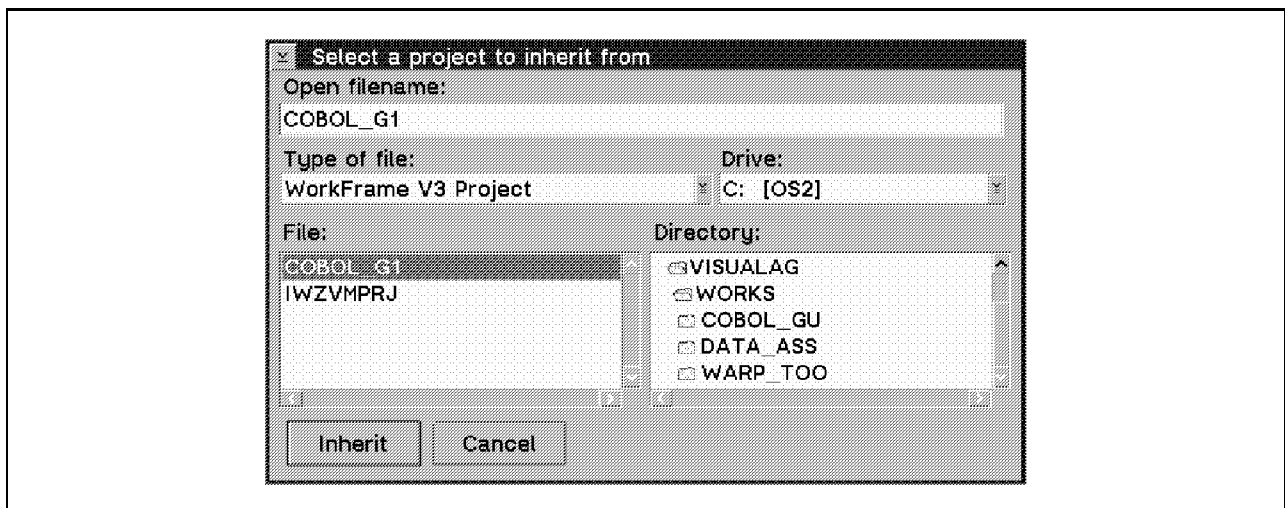


Figure 96. Select a Project to Inherit from Window

- Click on the **General** tab. The General page will appear (Figure 97 on page 52).

Type SALESDEP in the *Title* field.

Close the COBOL Project Settings Window. The **COBOL project** icon is renamed to SALESDEP.

Now your SALESDEP project should be set correctly.



Figure 97. SALESDEP - Settings General Page

7. To ensure that the project has been created successfully, you can open an OS/2 window and change the directory to d:\IBMCOBOL (d: is the drive where you installed IBM VisualAge for COBOL for OS/2). You will see a SALESDEP subdirectory. Now you can copy all the files under the SALESDEP directory of the diskette to d:\IBMCOBOL\SALESDEP by typing the following command:

```
copy a:\salesdep\*.* d:\IBMCOBOL\SALESDEP
```

8. Switch back to the OS/2 Desktop and double-click on the **SALESDEP** icon. The SALESDEP - Icon View window (Figure 98 on page 53) shows all the files required to run the application.

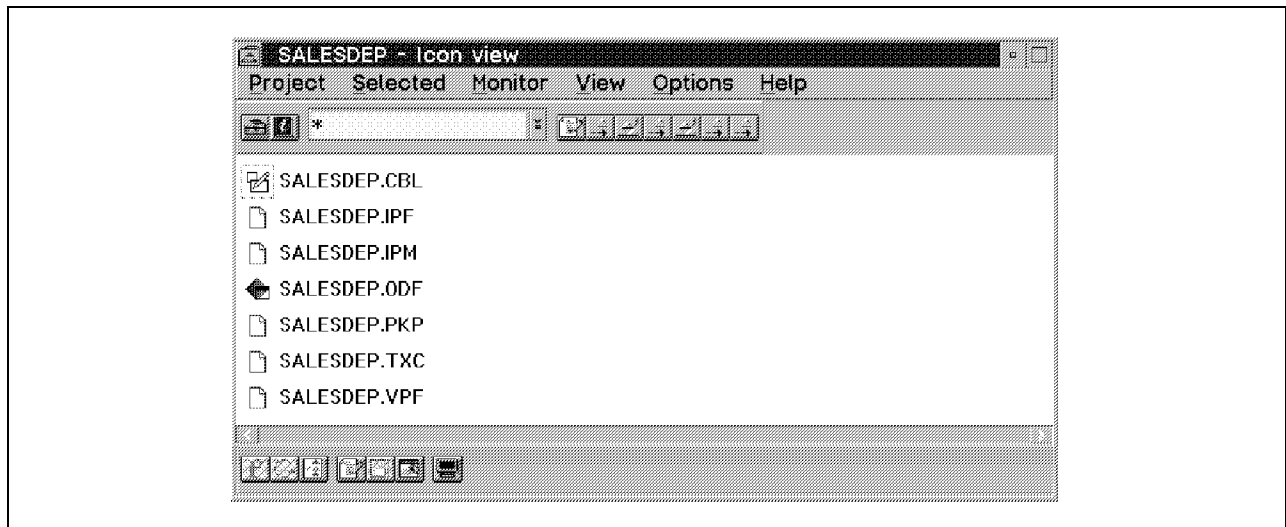


Figure 98. SALESDEP - Icon View Window (before building the project)

9. Because this application accesses data from a remote DB2 for OS/2 server, you have to rebind the application to build up a database access plan for your machine. We recommend that you rebuild the application. In the SALESDEP - Icon view window (Figure 98), set the compile option for the preprocessing of SQL. Select **Compile** from the Options pull-down menu. The GUI Compile: File Scope - IBM COBOL Compile Options window appears (Figure 99).

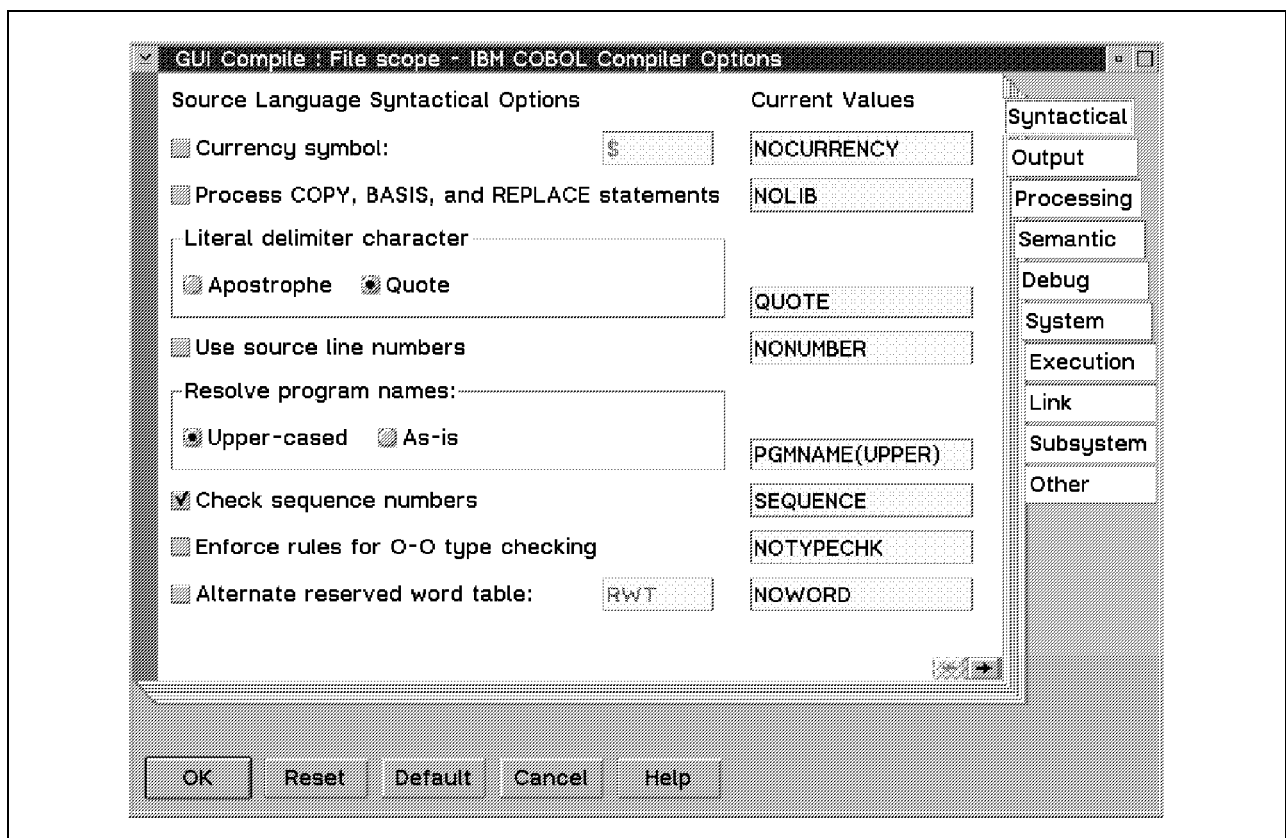


Figure 99. GUI Compile: File Scope—IBM COBOL Compiler Options Window

10. Scroll to the **Execution** tab (Figure 100 on page 54) and select **Generate optimized code**

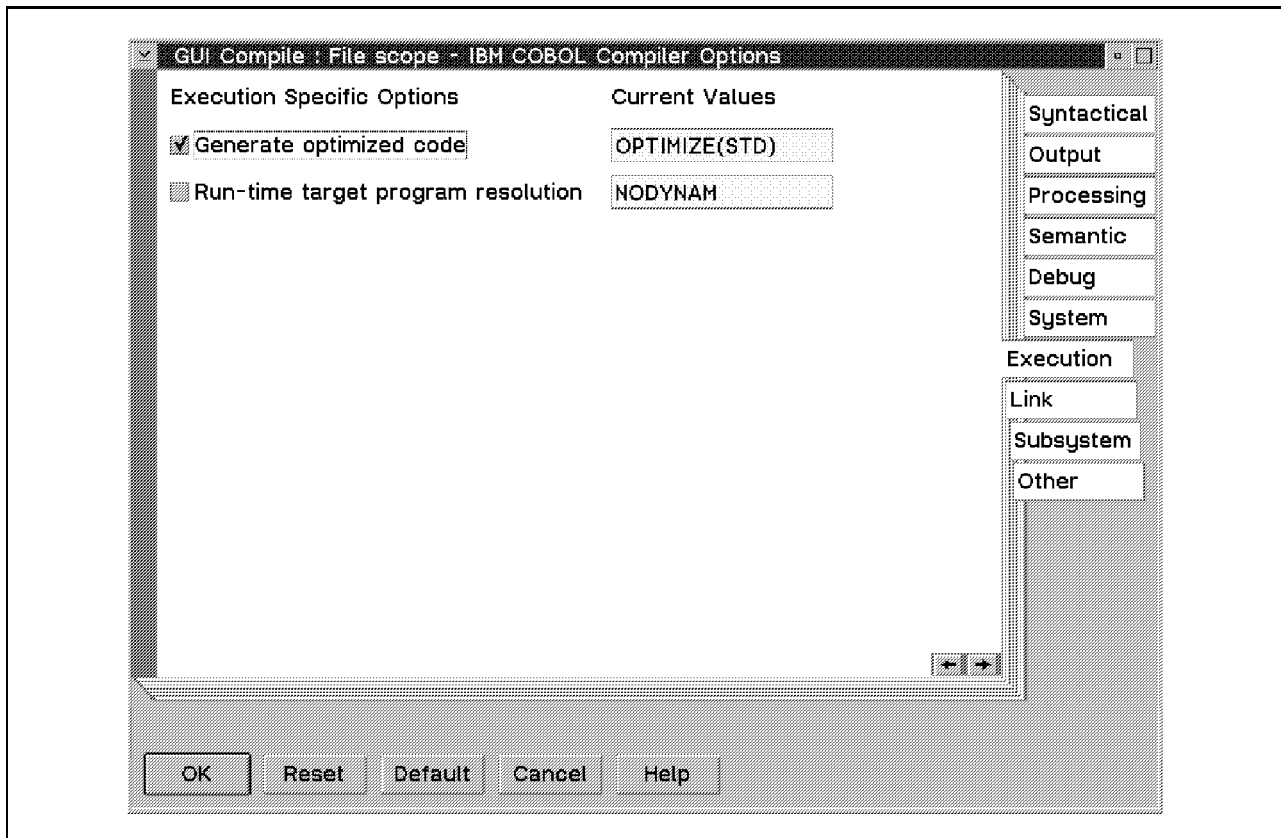


Figure 100. GUI Compile: File Scope - IBM COBOL Compiler Options Window—Execution Options

Optimize the code

Use OPTIMIZE to reduce the run time of your object program. Optimizations performed include the propagation of constants and the elimination of computations whose results are never used. Optimization might also reduce the amount of storage your object program uses. Because OPTIMIZE increases compile time, and can change the order of statements in your program, it should not be used when debugging.

In the following you will not be requested to set this option all the time, because our programs are too small to reach any success you can see using this option. But nevertheless you can set this option in each of your program.

To get to the Link page (Figure 101 on page 55), click on the **Link** tab.

Type `db2api.lib` in the *Enter library/object file name(s)* field to specify the required DB2 for OS/2 API library file.

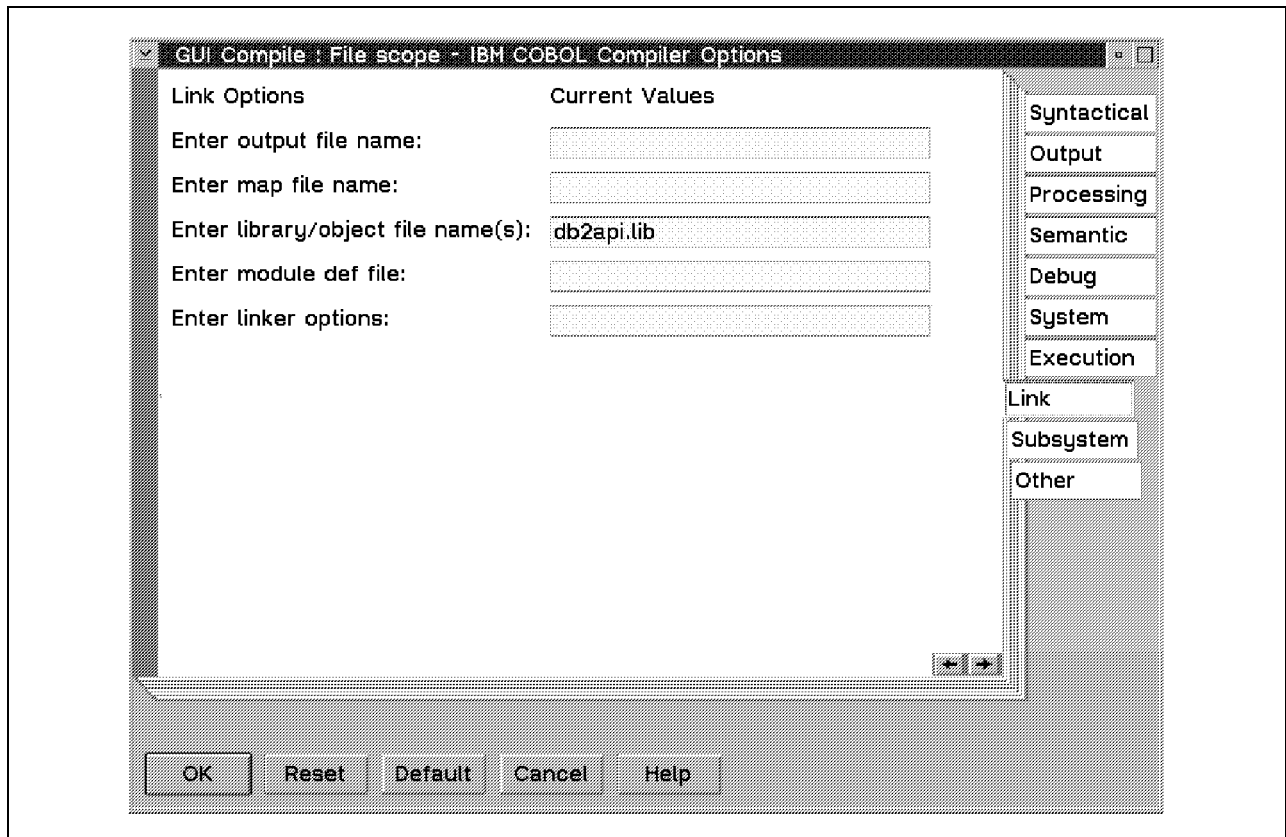


Figure 101. GUI Compile: File Scope - IBM COBOL Compiler Options Window—Link Options

Click on the **Subsystem** tab to get to the Subsystem page. Check the **Coprocess for SQL** check box.

Click on **Edit Options** and type

DATABASE SAMPLE BINDFILE PACKAGE

in the Edit SQL Coprocessor Options window that is coming up (Figure 102).

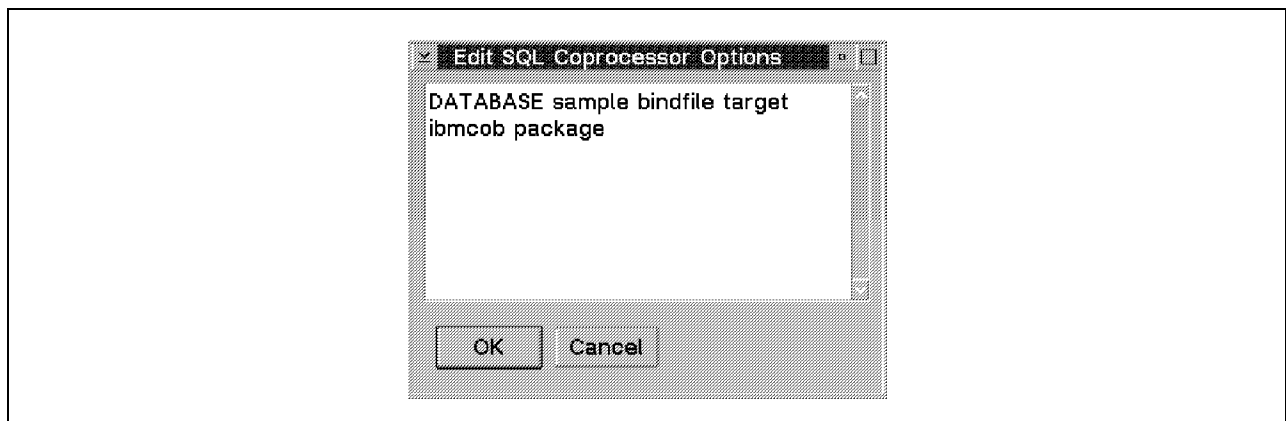


Figure 102. Edit SQL Coprocessor Options Window

Click on **OK** to quit this window. The specifications for the SQL coprocessor are now included in the Subsystem page (Figure 103 on page 56).

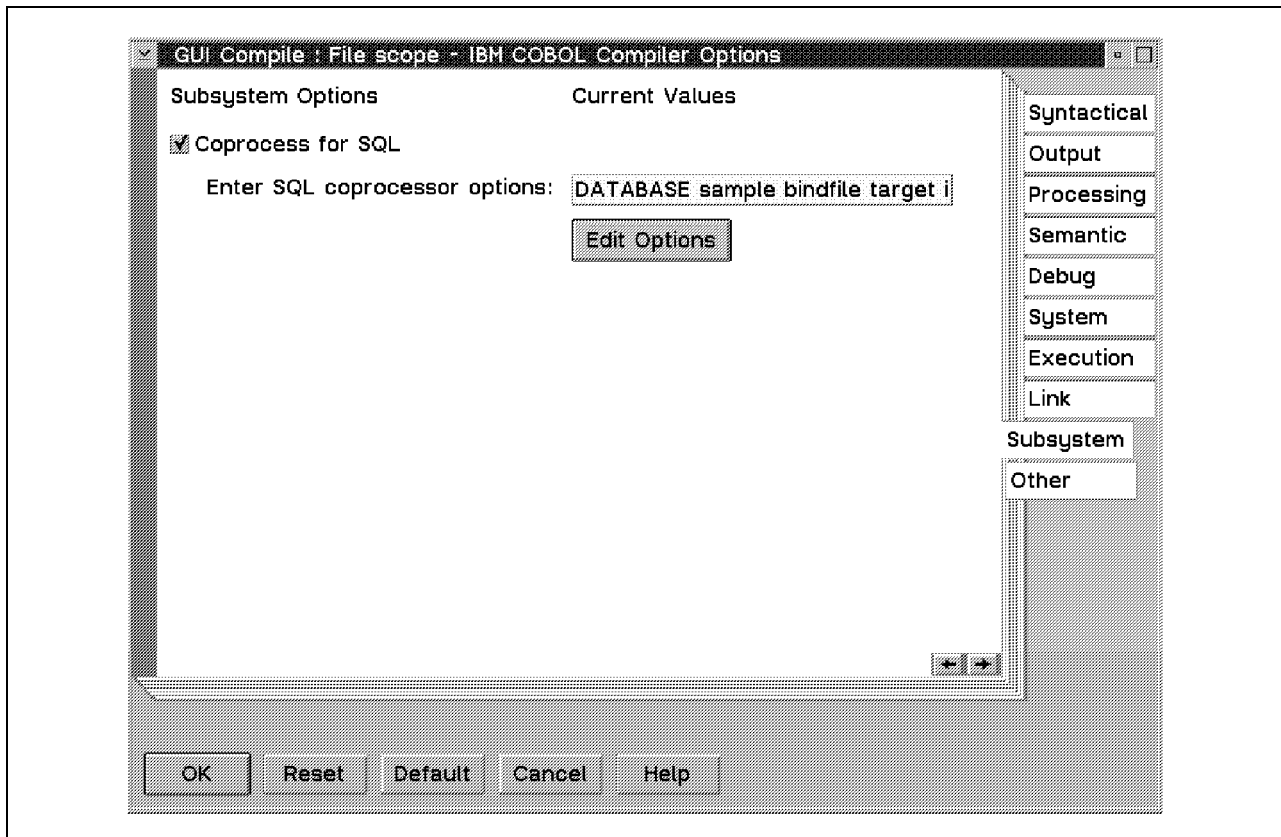


Figure 103. GUI Compile: File Scope - IBM COBOL Compiler Options Window—Subsystem Options

Click on **OK** to change the options.

11. To rebuild the application, from the Project pull-down menu in the SALESDEP - Icon view window (Figure 98 on page 53) click on **Build** and then select **Rebuild all** from the cascaded menu.

The system will start rebuilding the entire project by precompiling, compiling, and linking the sources. At the precompiling (for SQL statements) stage, the preprocessor needs to connect to the Sample database on the server. Therefore, if you have logged on the server with the authorized user ID, the preprocessor can connect to the Sample database and precompile all the SQL statements. If you have not logged on the database server, a Node Logon window will appear to ask you to log on to the database server node (Figure 104 on page 57). Type the appropriate user ID and password in the corresponding fields and click on **OK**. When you log on with the authorized user ID and password, the preprocessor will be able to connect to the Sample database and continue the SQL preprocess.

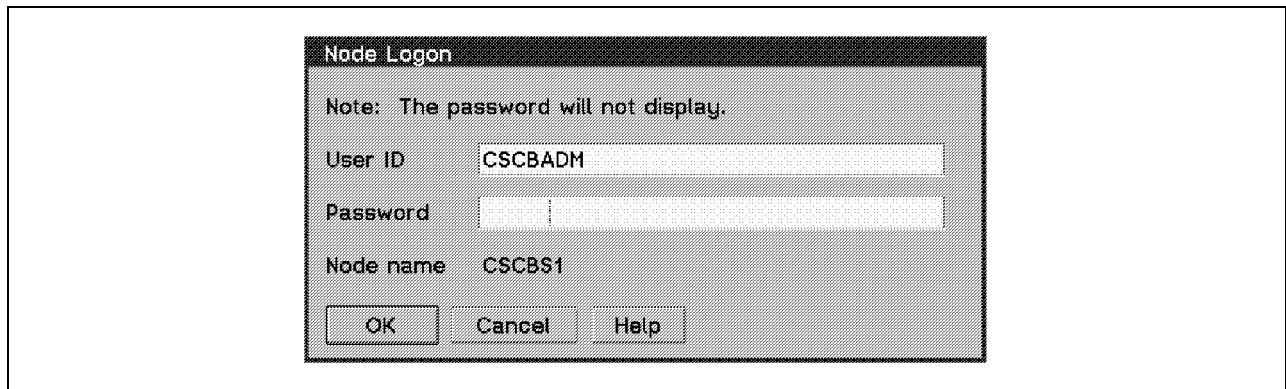


Figure 104. Node Logon Window

Note

If you are already logged on locally at DB2 with the user ID named USERID and PASSWORD as the password and you try to compile the program that accesses DB2 on the server you get an error message while compiling because you did not define USERID as a valid user ID in your LAN.

In the case you are logged on locally, log off first and log on again now in the LAN as described above.

The build process now passes through various stages. When you receive messages indicating that the build operation succeeded, as shown in Figure 105, you may begin running the application.

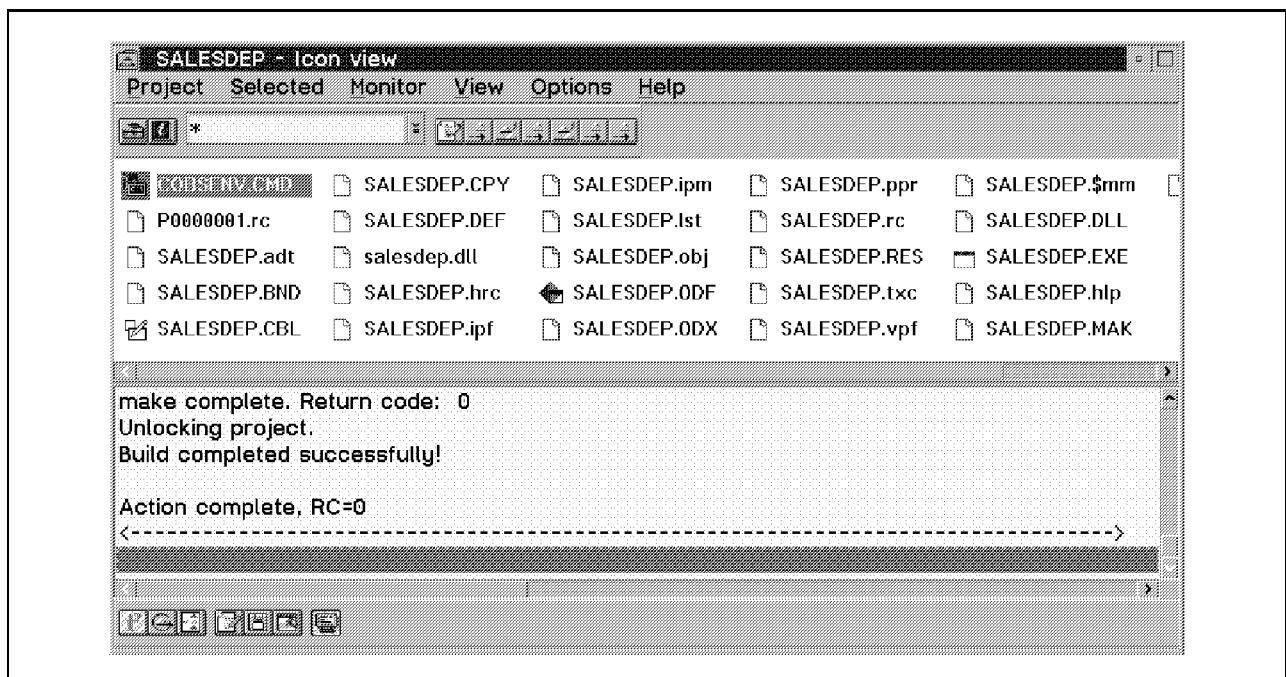


Figure 105. SALESDEP - Icon View Window (after building the project)

12. Select **Run** from the Project pull-down menu to execute SALESDEP, and the New Salary for Sales Department window will appear (Figure 106 on page 58).

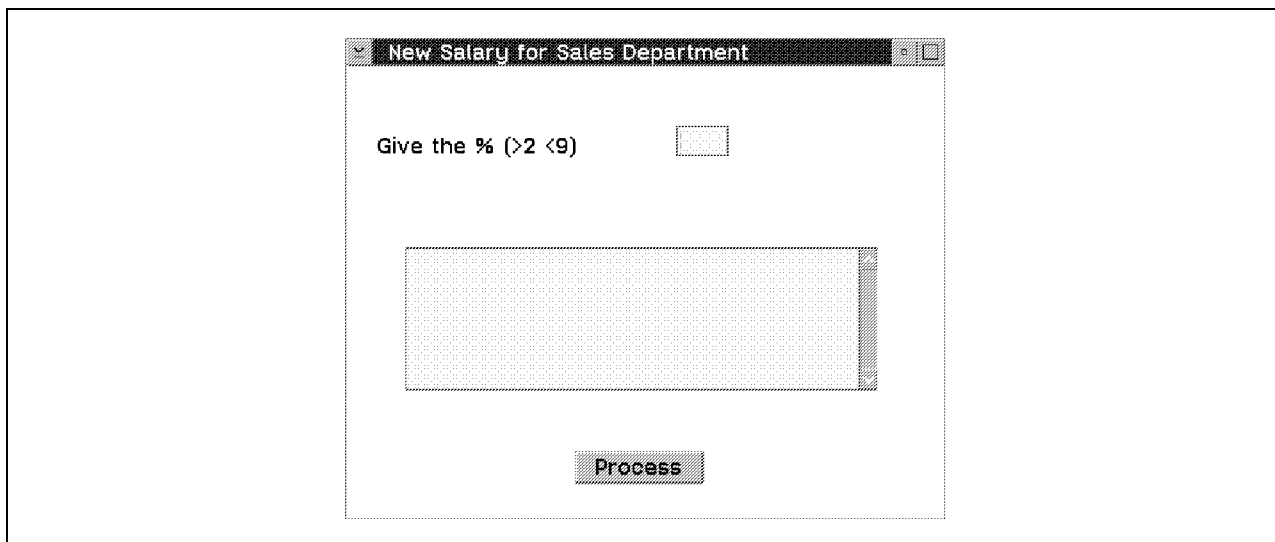


Figure 106. New Salary for Sales Department Window

13. The application begins, and you can enter a number for the percentage increase over the original salary. The application will access the SAMPLE database on the CSCBSVR server machine. The results appear in the list box of the window labeled New Salary for Sales Department (Figure 107).

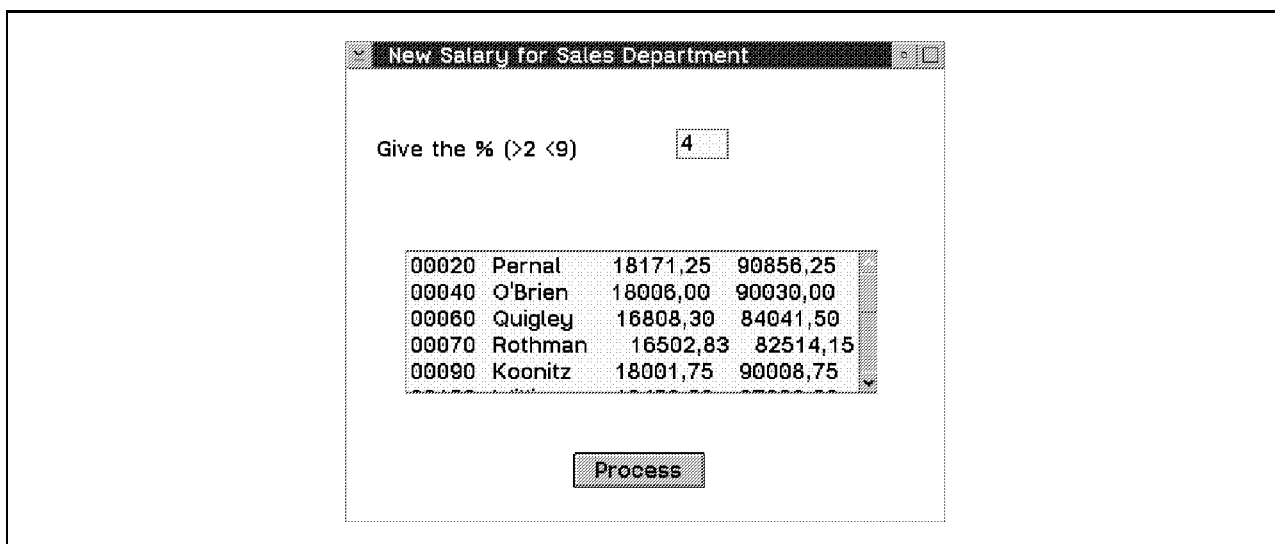


Figure 107. New Salary for Sales Department Window: Original and New Salaries

Now you have finished executing the IBM VisualAge for COBOL for OS/2 application in an OS/2 client/server environment.

Chapter 2. OS/2 LAN with CICS and DB2

This chapter describes how to set up an OS/2 LAN-based client/server environment that includes the Customer Information Control System (CICS) for OS/2 in addition to the products described in Chapter 1, "OS/2 Local Area Network with DB2" on page 1.

Figure 108 shows an overview of the environment.

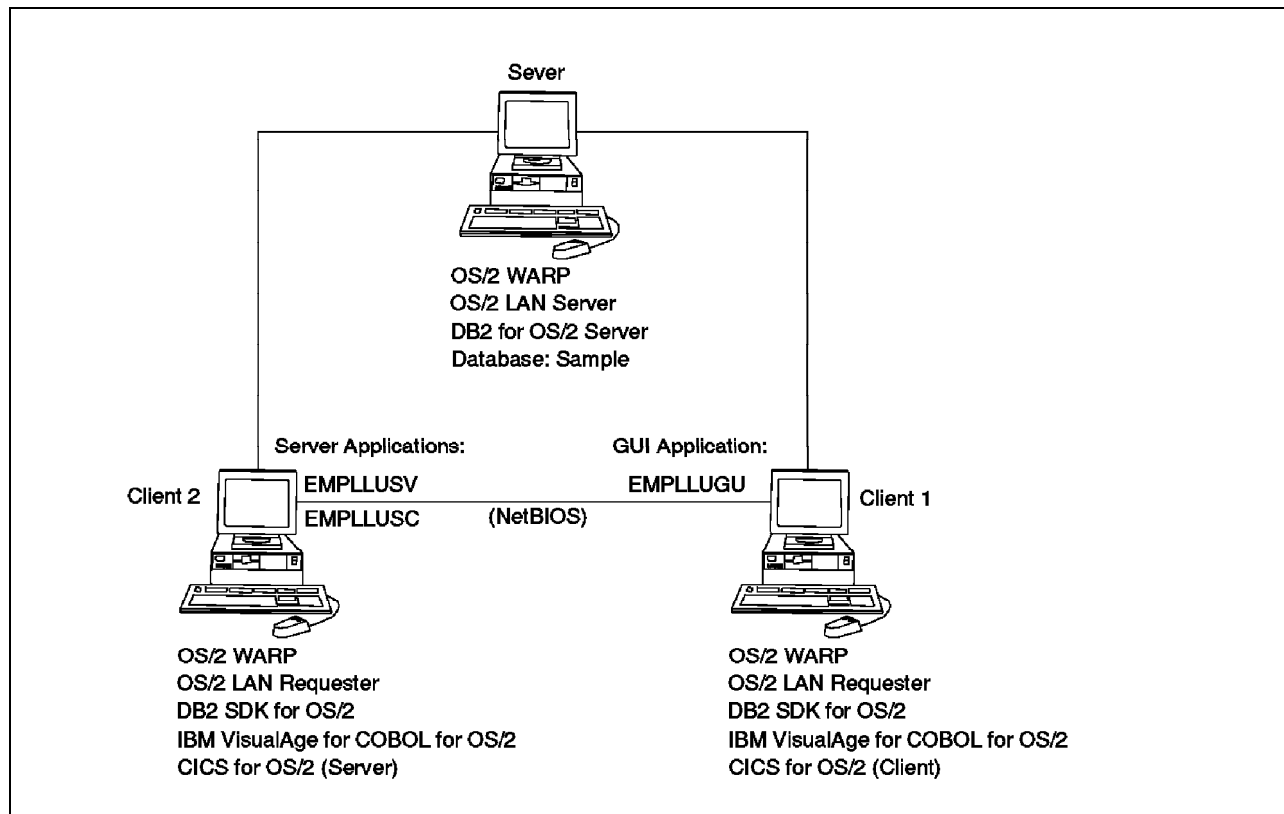


Figure 108. OS/2 LAN-Based Environment Including CICS

This environment is based on the setup described in Chapter 1, "OS/2 Local Area Network with DB2" on page 1. The comments we added there apply here as well.

CICS Client

You must install CICS Client on each workstation where you want to execute a program that calls a CICS program through an external call interface (ECI) call. A CICS program is one that contains at least one "EXEC CICS" statement and it must run on a workstation where CICS for OS/2 is installed. A COBOL GUI client program must not be a CICS program. You must always use ECI to invoke the CICS environment from a GUI program. To develop (compile and link) such a GUI ECI client program you do not need a CICS library (CICS Client or CICS for OS/2) on your workstation.

CICS for OS/2

You must install CICS for OS/2 on every workstation where you want to either develop or execute a CICS program.

CICS for OS/2 does not include CICS Client. Therefore, if you want to execute a GUI client program on the same workstation as the CICS program, you must install CICS Client separately.

2.1 CICS for OS/2

CICS for OS/2 is a member of the CICS family. It creates a CICS environment under OS/2 and provides CICS server support. It communicates with other CICS platforms and client workstations.

The main communication mechanisms used by CICS for OS/2 in a client/server environment are:

- External Call Interface. ECI allows non-CICS programs to call CICS programs in a CICS server and pass data via COMMAREA.
- External Presentation Interface (EPI). EPI allows a non-CICS program to be viewed as a 3270 terminal by a CICS server system to which it is connected.
- External Transaction Initiation (ETI). ETI allows non-CICS programs to access and update CICS resources by initiating CICS transactions.

In this chapter, the program that you execute in the CICS client machine uses the ECI function to call a CICS program in the CICS server and run it as a subroutine.

Begin by setting up a CICS client/server environment that includes a database server, a CICS server, and a CICS client. You then execute a GUI client program on the CICS client machine that invokes a CICS COBOL program on the CICS server machine.

To utilize the environment set up in the previous chapter, install CICS for OS/2 on the development machine (Section 2.1.1), install CICS Client for OS/2 on the client machine (Section 2.2), and use the DB2 for OS/2 server machine you already have.

2.1.1 CICS for OS/2 Installation

This section describes the steps to install CICS for OS/2 on the development machine that will be used as the CICS server of the client/server environment.

To install CICS for OS/2, complete the following steps:

1. Insert the Transaction Server CD-ROM or diskettes into the drive, change the directory to the \US directory, and type:

```
install
```

to start the installation.

The Installation window with the CICS Logo appears as shown in Figure 109 on page 61.

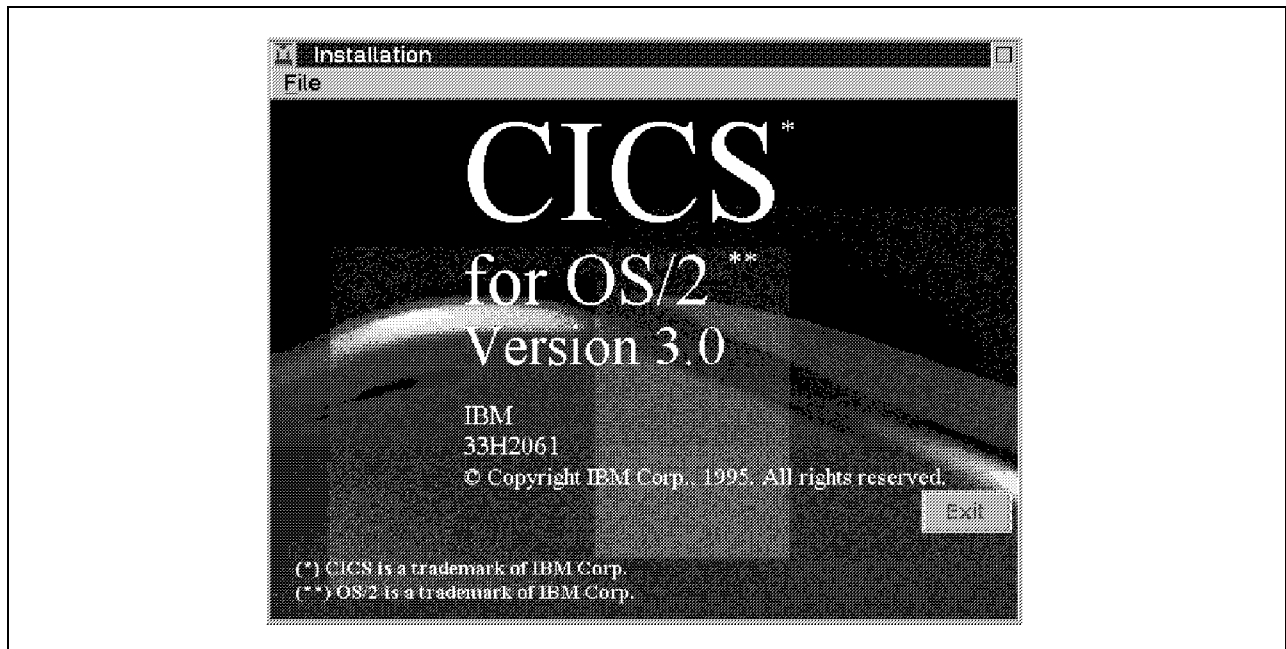


Figure 109. Installation Window: CICS for OS/2

2. The Install window (Figure 110) appears after the Installation window. Choose the default option **Update CONFIG.SYS**, and click on **OK** to continue with the installation.

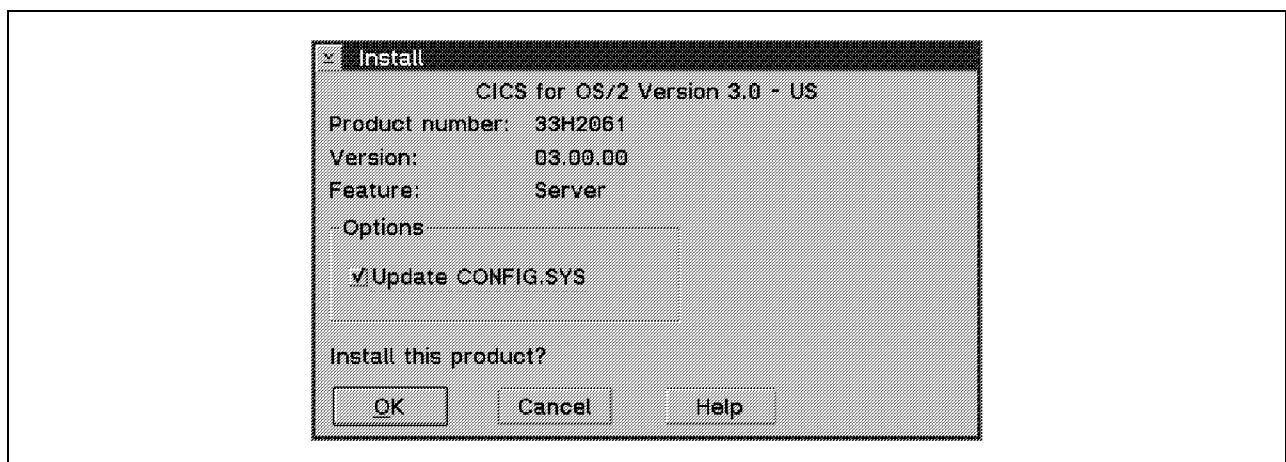


Figure 110. Install Window with CICS Identification

3. An Install—Directories window appears from which you can select the components that you want to install (Figure 111 on page 62).

Select the following components:

CICS for OS/2 Run-Time System
CICS for OS/2 Development
CICS for OS/2 COBOL Sample Code
CICS for OS/2 On-Line Manuals

Set the directory in which you want to install CICS for OS/2. You can click on **Disk Space...** to check the space available on your disk drive.

Click on **Install**.

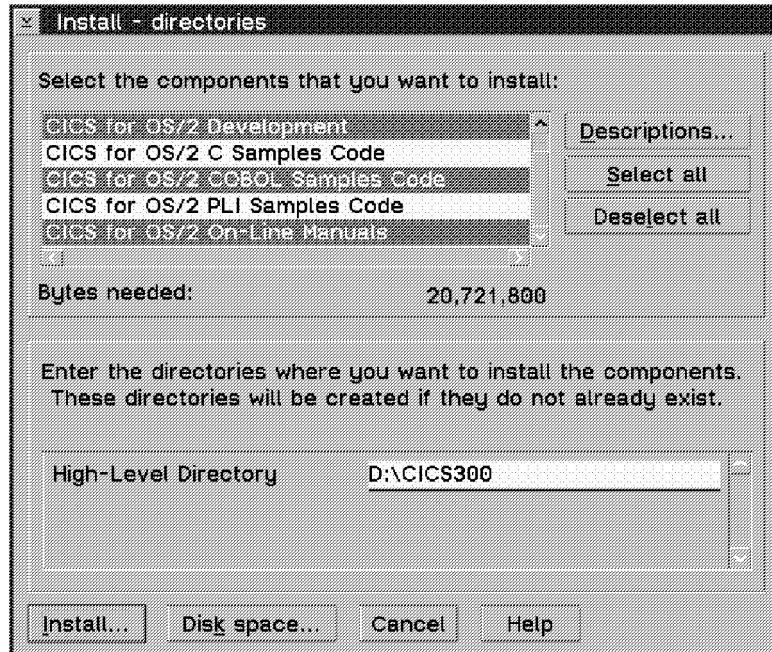


Figure 111. Install - Directories Window

4. While the system is installing CICS for OS/2, the Install-Progress window displays the progress of the installation (Figure 112).

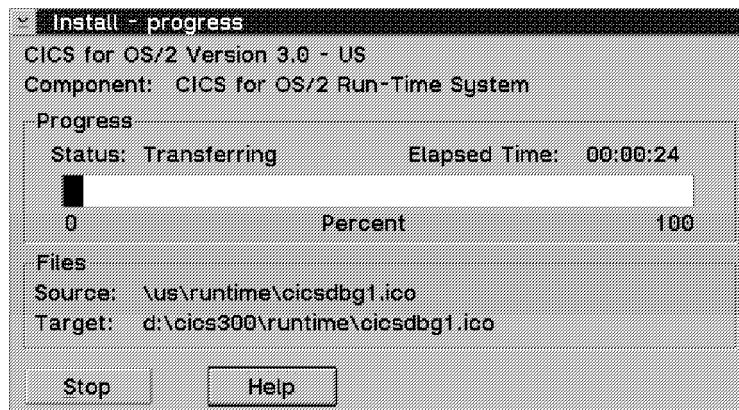


Figure 112. Install - Progress window

5. The Install - System Configuration window appears when the installation is nearly finished. In the window, you can set the SYSID and APPLID for your system.

SYSID is the local name of the CICS for OS/2 system. The default is CICS. APPLID is the NetBIOS name of your CICS for OS/2 machine. The default is CICSOS2. Each CICS client connected to this server must specify this APPLID name in the *ServerName* field of its configuration file (CICSCLI.INI). Because the APPLID must be unique in your LAN environment, you should use other names for SYSID and APPLID, instead of the default settings. Figure 113 on page 63 is an example of these two entries. After you type the names in the *SYSID* and *APPLID* fields, click on **OK**.

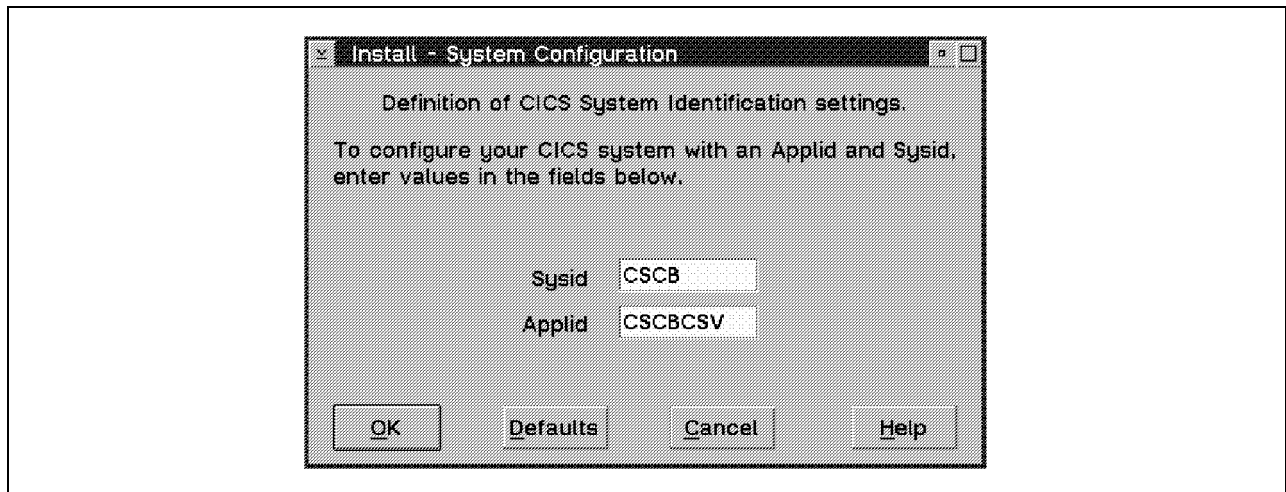


Figure 113. Install—System Configuration Window

6. When the installation is complete the Installation and Maintenance window appears (Figure 114). Click on **OK**. You are now back at the OS/2 prompt. Reboot the system in order to activate the change modified by the installation.

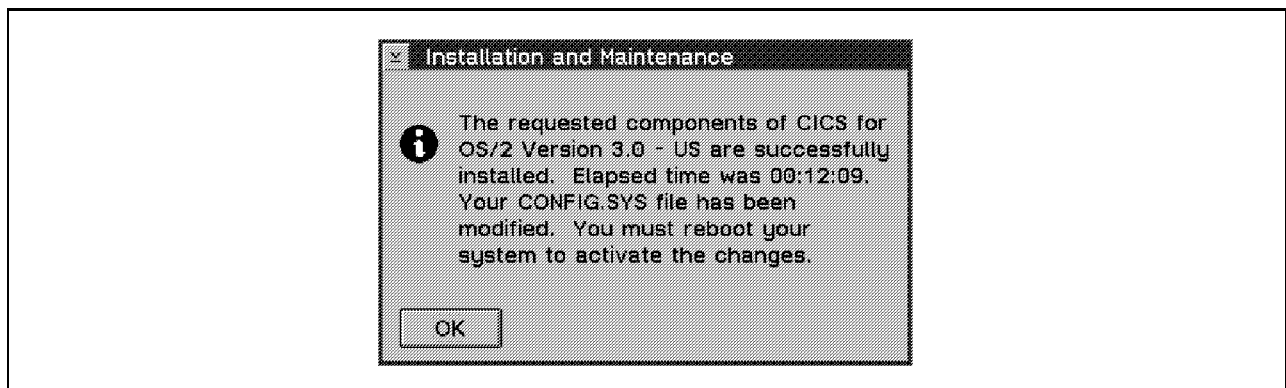


Figure 114. Installation and Maintenance Window Showing Completion

7. After rebooting the system, double-click on the **CICS for OS/2 Version 3.0** icon on the Desktop. The CICS for OS/2 Version 3.0 Icon View window appears (Figure 115).

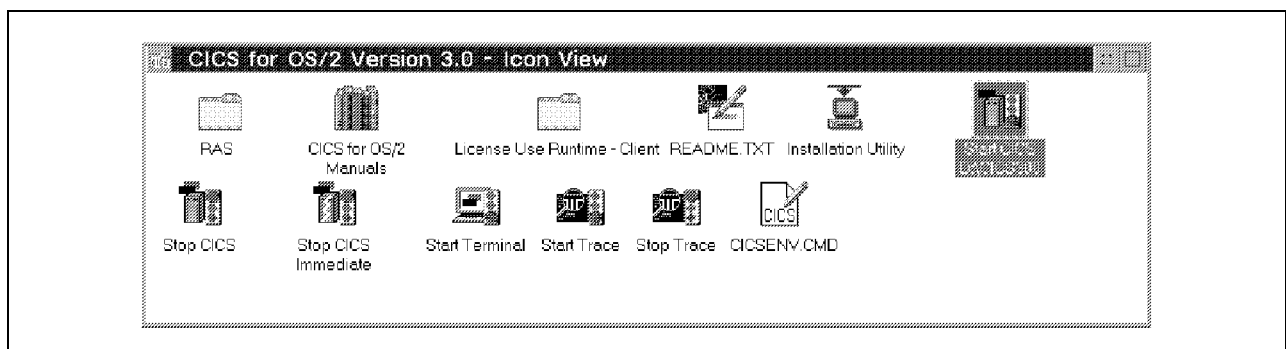


Figure 115. CICS for OS/2 Version 3.0 - Icon View Window

8. Double-click on the **Start CICS d:\CICS300** icon (d: is the drive where you install the CICS for OS/2). Start CICS d:\CICS300 window appears containing

several startup messages. If you are starting CICS for the first time, the License Information window appears. Press **Continue** in the License Information window. Two windows appear. One is the CICS for OS/2 V3.0 window, which displays the CICS message logs (Figure 116).

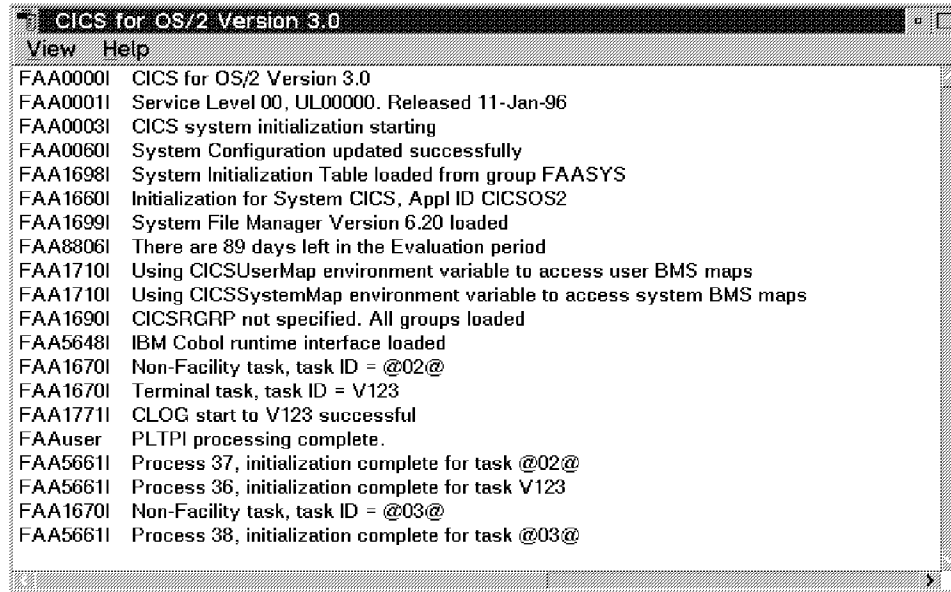


Figure 116. CICS for OS/2 V3.0 Window, Showing Message Logs

The other window is labeled CICS for OS/2 Terminal V123, which is the CICS terminal you will work with (Figure 117 on page 65).

9. In Figure 117 on page 65, you see the message, *Enter to continue*. In the CICS environment, the Enter key is defined as the right-hand Ctrl key on the keyboard. Press Enter to continue the CICS startup process.



Figure 117. CICS for OS/2 Terminal V123 Window

10. The CICS Sign-on panel appears and waits for your sign-on. Type `sysad` as the user ID and password (Figure 118 on page 66). User `sysad` is the predefined CICS system administrator. You will use the `sysad` to do all the administration jobs. You can change the ID if you want to use another name instead of `sysad`.

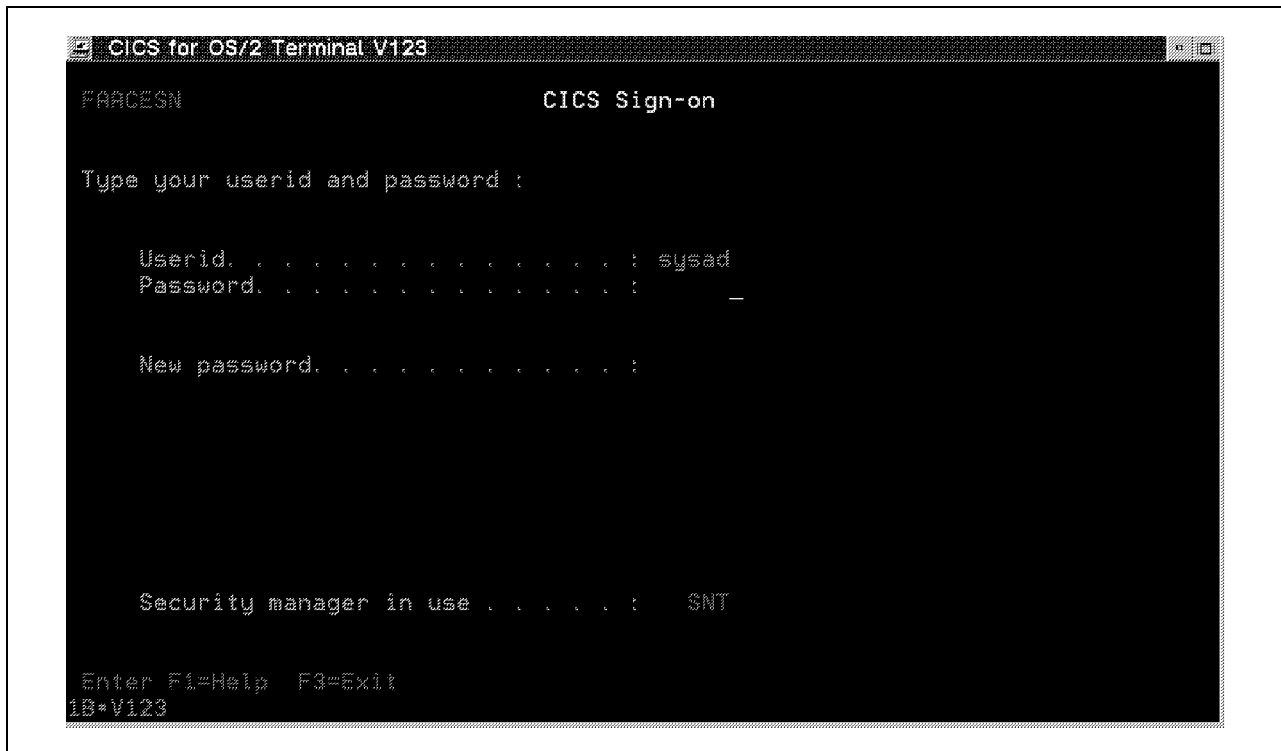


Figure 118. CICS Sign-on Panel

11. Press the Enter key. You will see the message *Signon completed successfully* displayed. Press Esc to clear the screen. You are now signed on to your CICS for OS/2 system with the administrator authority and ready to execute CICS transactions.
12. To build up the communication between CICS server and client, you must change certain settings in the System Initialization Table (SIT).
Type *CEDA* on the CICS Terminal and press Enter. The Resource Definition Online panel appears (Figure 119 on page 67).

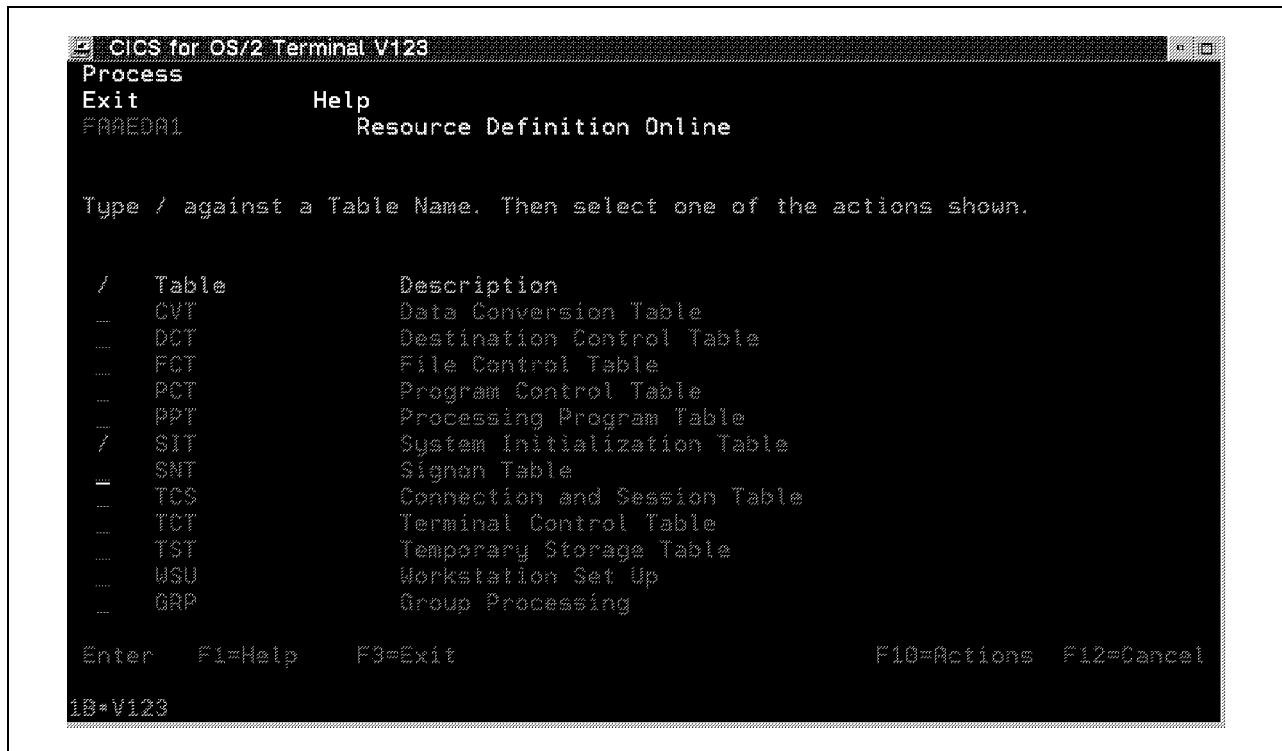


Figure 119. Resource Definition Online Panel

13. Move the cursor to the SIT entry, type /, and press Enter. The System Initialization Table panel will be displayed (Figure 120).

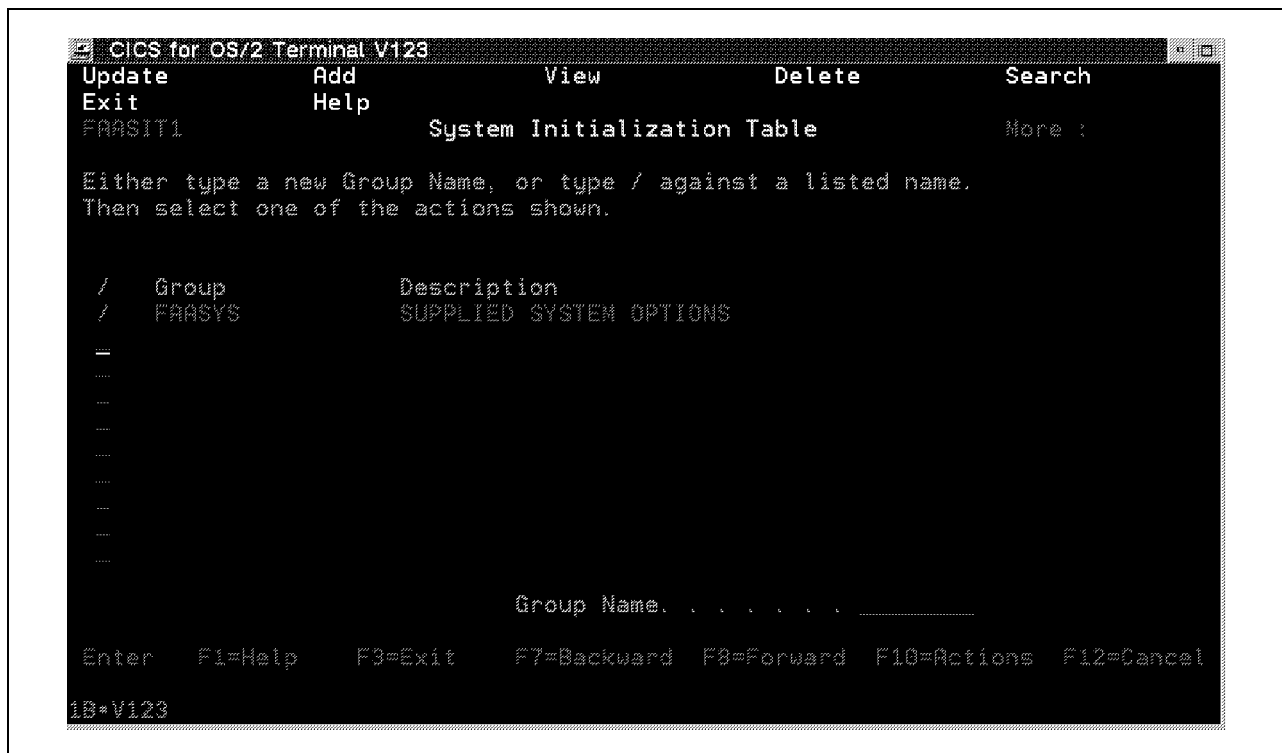


Figure 120. System Initialization Table Panel

14. Press F8 to get to the next page of the SIT panel. You will see the local system ID and the local system Appl ID. The local system ID is the SYSID

and the local system Appl ID is the SYSID you selected when you installed CICS for OS/2 (Figure 121 on page 68). If you want to change the SYSID and APPLID, you can modify them in this panel. Here, you must change the settings of NetBIOS support. Move the cursor to the *NetBIOS Listener Adapter* field. Type 0. Move the cursor to *Maximum NetBIOS Systems* field. Type 10. Press Enter to make the changes. A message will be displayed to inform you of the changes that have been made. Press Enter to clear the message and press **F3** to return to the Resource Definition Online Panel.

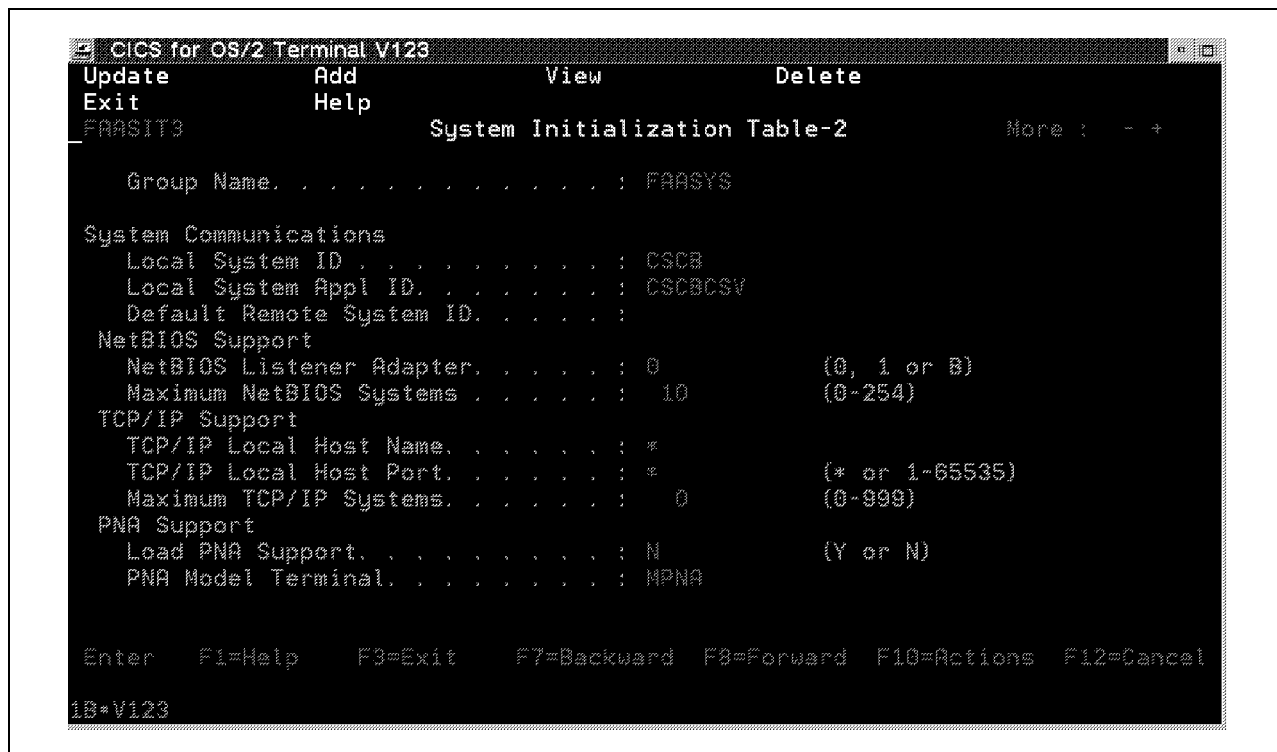


Figure 121. CICS for OS/2 Terminal V123 Window—System Initialization Table-2 Panel

15. Stop CICS for OS/2 by double-clicking **Stop CICS** or typing CQIT and pressing Enter in the CICS terminal.
16. Start CICS for OS/2 again. The following messages in the CICS log window (Figure 122 on page 69) appear:
NetBIOS Listener starting for CSCBCSV on adapter 0

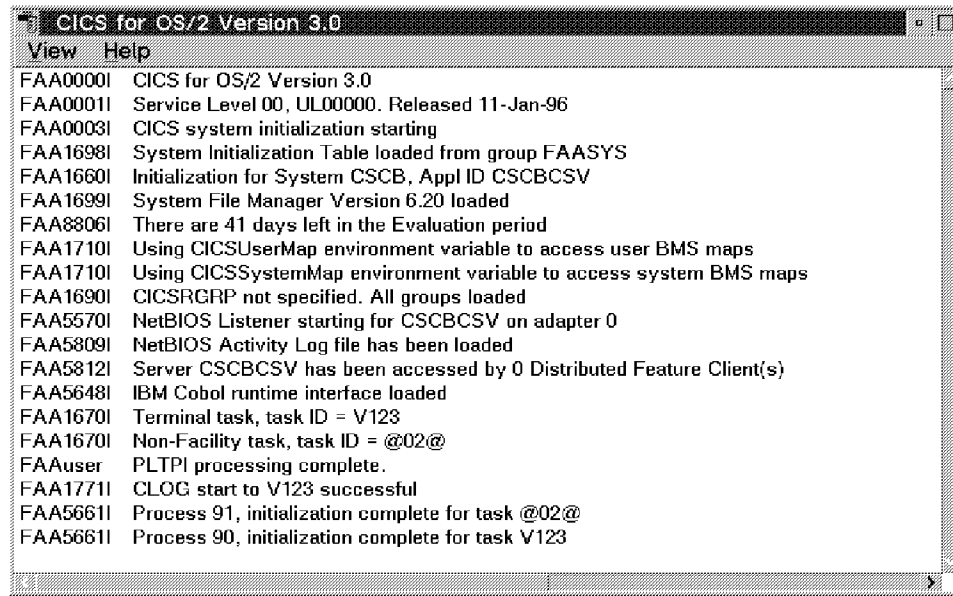


Figure 122. CICS for OS/2 Version 3.0 Window—Log, Showing Completion Message

After the restart, the CICS server machine is ready to communicate with CICS clients over NetBIOS.

2.2 OS/2 CICS Client

CICS for OS/2 provides CICS server support for IBM CICS Clients. This means that a single CICS for OS/2 system can provide CICS server functions to multiple workstations that are running as IBM CICS Clients.

The family of IBM CICS Clients comprises workstation products that communicate with the family on CICS application servers. CICS clients add the advantages of client/server operation to your transaction processing. A CICS client can communicate with multiple CICS servers.

The CICS clients can communicate with a server by three mechanisms:

- The *External Call Interface* enables the design of new applications to be optimized for client/server operation, with the business logic on the server and the presentation logic on the client.
- The *External Presentation Interface* enables modern technologies, such as graphical or multimedia interfaces, to be used with traditional 3270 CICS applications.
- *3270 terminal and printer emulation* provides CICS 3270 emulation for CICS servers to which the client is connected.

ECI Calls

The ECI allows a non-CICS application to call a CICS program in a CICS server. The non-CICS application does not issue any CICS commands itself. The CICS commands are issued by the called program running in the server.

The application can be connected to several servers at the same time and can have several program calls outstanding at the same time as well.

The CICS program cannot perform terminal I/O, but can access and update all other CICS resources.

The server program is called by a program link call. Program link calls cause a CICS program to be executed on a CICS server. Program link calls can be either synchronous or asynchronous. For asynchronous calls, it is the responsibility of the calling application to solicit the reply using one of the reply solicitation calls.

If a call is synchronous, the application waits until the called program has finished.

If a call is asynchronous, the application regains control without reference to the completion of the called program. The application can ask to be notified when the information becomes available. It must use a reply solicitation call to determine the outcome of the asynchronous request.

The COBOL program's transaction assistant supports only synchronous calls.

2.2.1 CICS Client Installation

This section describes the installation of CICS Client for OS/2 on the client machine. CICS Client for OS/2 supports NetBIOS, TCP/IP and APPC communication protocols. Because you use the same environment set up in Chapter 1, you only need to install CICS Client for OS/2 on your client machine.

To install CICS Client, follow these steps:

1. Insert the Transaction Server CD-ROM or diskettes into the drive. Change to the correct drive, and change the directory to `\clients\os2`. Type `install` to start installing CICS. The CICS logo window will appear (Figure 123 on page 71).

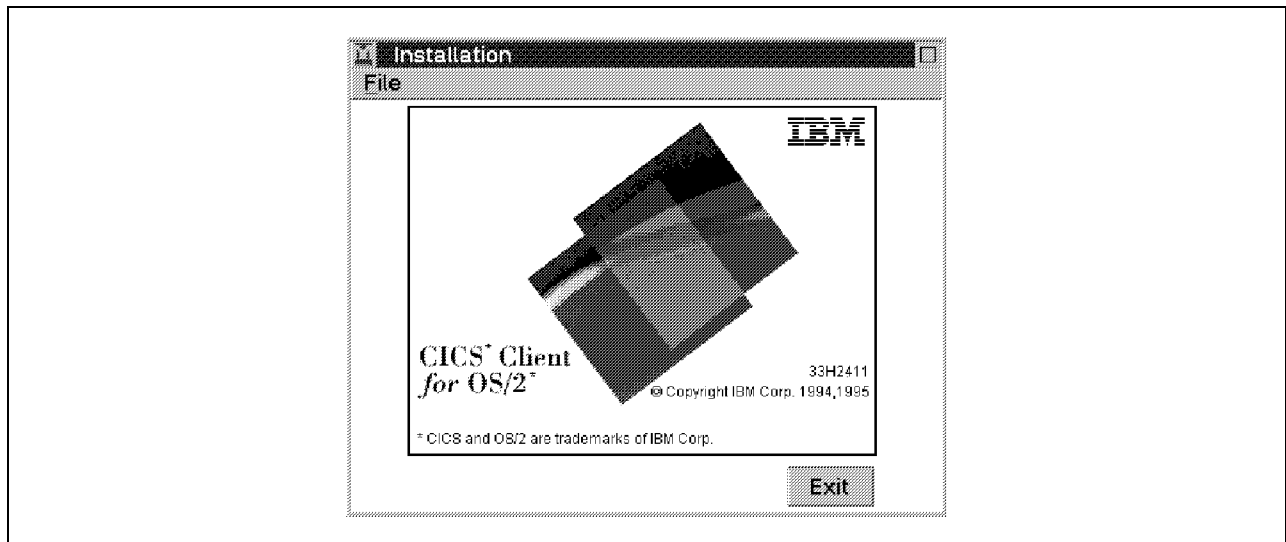


Figure 123. CICS Logo Window for CICS Client for OS/2

Then the Instructions window appears (Figure 124).

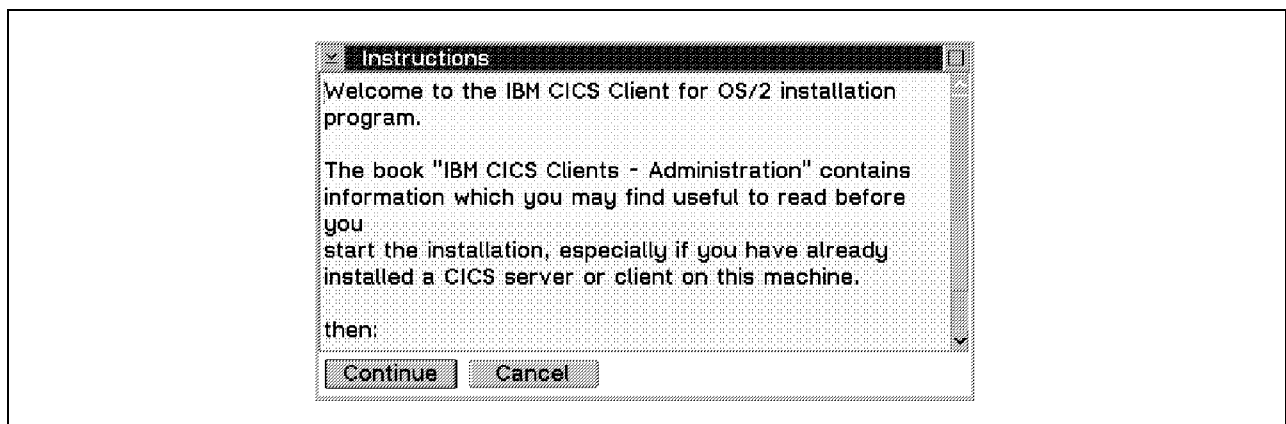


Figure 124. Instructions Window for CICS Client Installation

2. Click on **Continue** and the Install window will be displayed as shown in Figure 125.

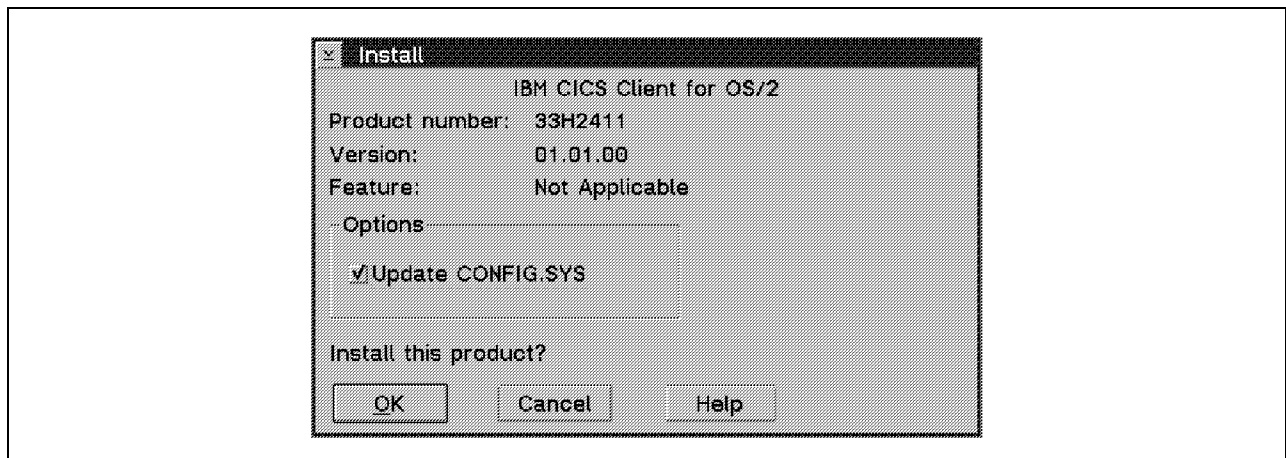


Figure 125. Install Window for CICS Client for OS/2

3. Click on **OK** to continue the installation. An Install - directories window appears from which you can select the components you want to install (Figure 126 on page 72).

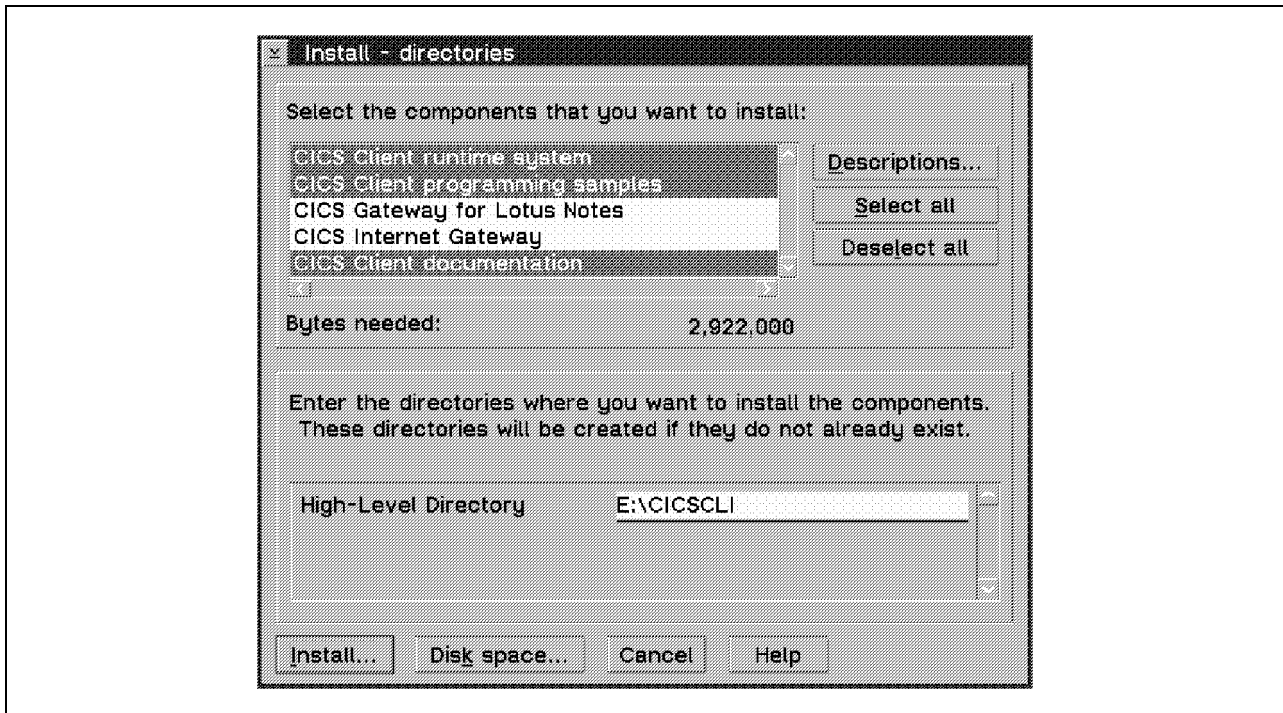


Figure 126. Install - Directories Window for CICS Client

Select the following components listed from the list box:

CICS Client runtime system
CICS Client programming samples
CICS Client documentation

Select the drive and directory in which you want to install CICS Client. Click on **Install**. The Install - Progress window is displayed to show the progress of the installation.

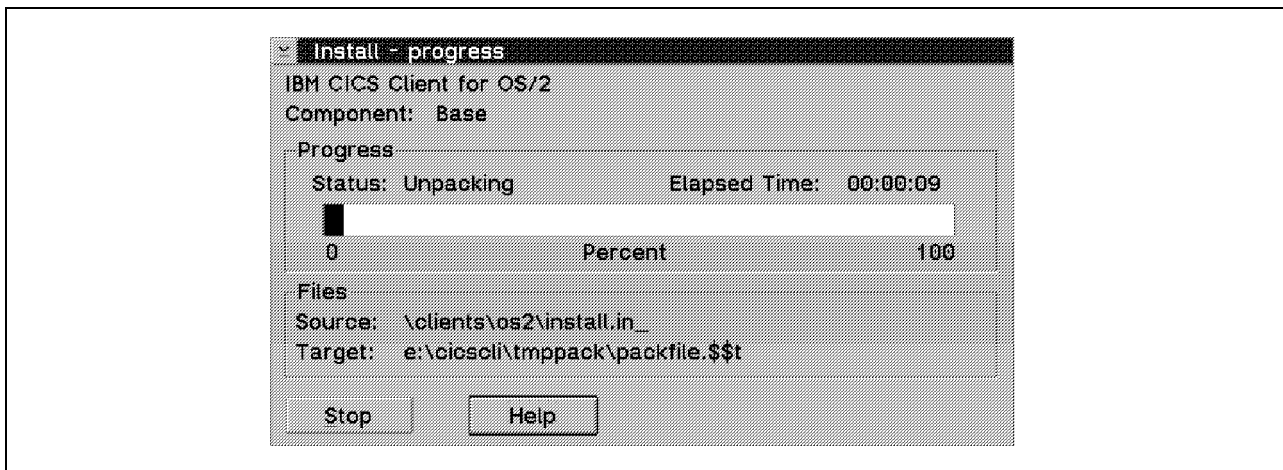


Figure 127. Install—Progress Window for CICS Client

4. When the installation is nearly finished, the IBM CICS Client for OS/2 - Language selection window will appear (Figure 128 on page 73). Select the language you are going to use and click on **OK**.

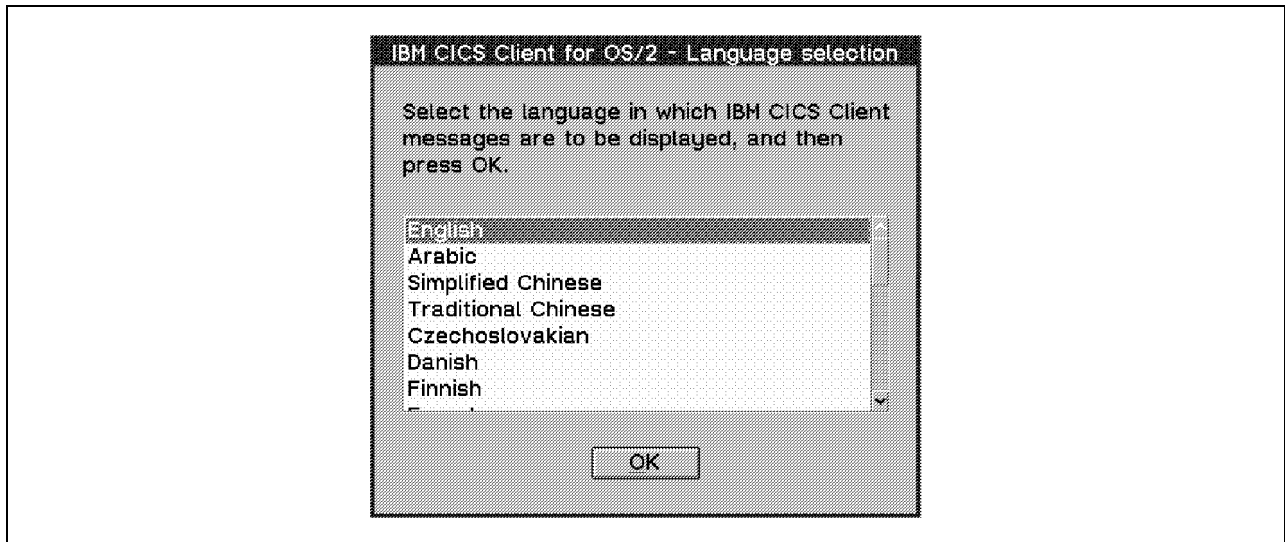


Figure 128. IBM CICS Client for OS/2 - Language Selection Window

5. The CICS Client for OS/2 - Question window appears (Figure 129). Click on **Yes** to see what has been installed. README.TXT is then displayed. Review the file and then close the window.

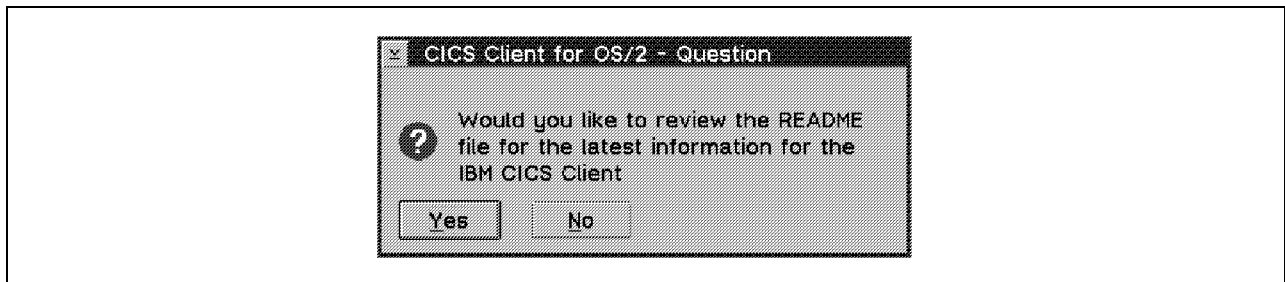


Figure 129. IBM CICS Client for OS/2 - Questions Window

6. When the installation is finished, the Installation and Maintenance window appears (Figure 130). Click on **OK**.

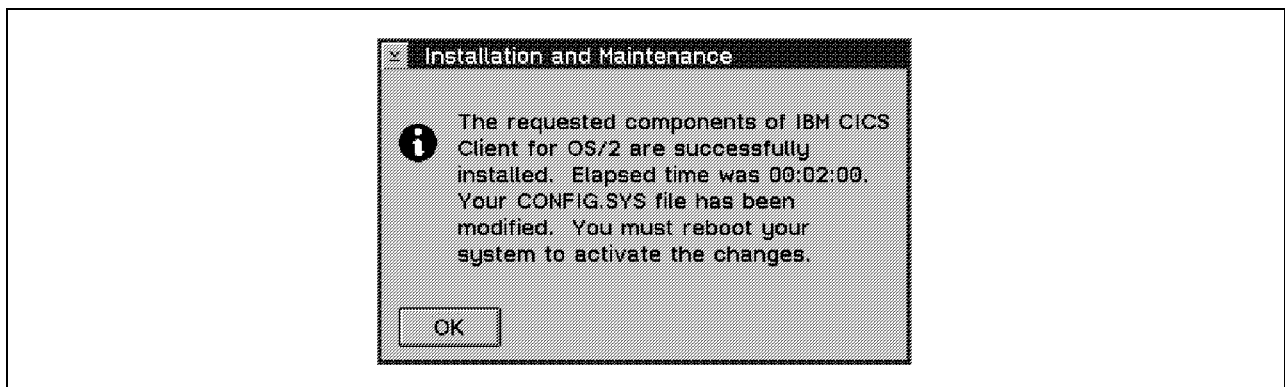


Figure 130. Installation and Maintenance Window

7. In order to connect to the CICS for OS/2 server machine you have installed, you have to modify the settings in the CICSCLI.INI file. The file is under the d:\CICSCLI\BIN directory and it defines the settings for the CICS client and the connected servers. The file contains some statements in the server section, such as:

```
Server=CICSNETB
:
Protocol=NETBIOS
NetName=CICSOS2
:
```

The NetName defined here should be the same as the APPLID you use on the CICS server machine. CICSOS2 is the default name. If you use another name for APPLID when you installed the CICS for OS/2 on your server machine, you must change the value of the NetName into the APPLID you use on the server machine. For instance, if you specified CSCBCSV in the APPLID field as shown in the Figure 113 on page 63, you have to type NetName=CSCBCSV in this file. After saving the change, the setting of the CICS Client machine should be correct.

8. The IBM CICS Client for OS/2 folder appears on the Desktop. Shut down and restart the machine. CICS Client is then ready for use.

You can now compile and execute the client programs in the CICS client/server environment.

CICSCLI

The CICS Client includes a tool named CICSCLI with which you can check the connection to the server. The underlying protocol and the definitions of the server are specified in the file CICSCLI.INI in the directory E:\CICSCLI\BIN where E is the drive where the CICS Client is installed. Typing CICSCLI in the command line in an OS/2 window gives you an instruction naming all choices you have using this command.

The command CICSCLI /L provides a list of the active server connections. This command is useful to test the connection.

With the command CICSCLI /S=CICSNETB you start the client and connect to the server named CICSNETB, and with CICSCLI /X you close the client.

2.2.2 Using OS/2 CICS Client

Once the installation of CICS Client is complete, add a user ID for this first user. Next, complete the following steps.

1. Start the CICS Server on the server machine.
2. Start the CICS on the client machine by double-clicking on the appropriate icon on the IBM CICS Client for OS/2 folder.
3. Start the CICS terminal by activating the **Start Terminal** icon.
4. When you are using CICS Client for the first time, you must log on as SYSAD in the *User ID* field and the password SYSAD in the *Password* field. Press Enter.
5. The CICS Terminal window appears. Type ceda in this window and press the Ctrl key to exit this window.

6. The next CICS Terminal window shows the Resource Definition Online (Figure 131 on page 75). Move the cursor to the SNT entry, type /, and press the Ctrl key.

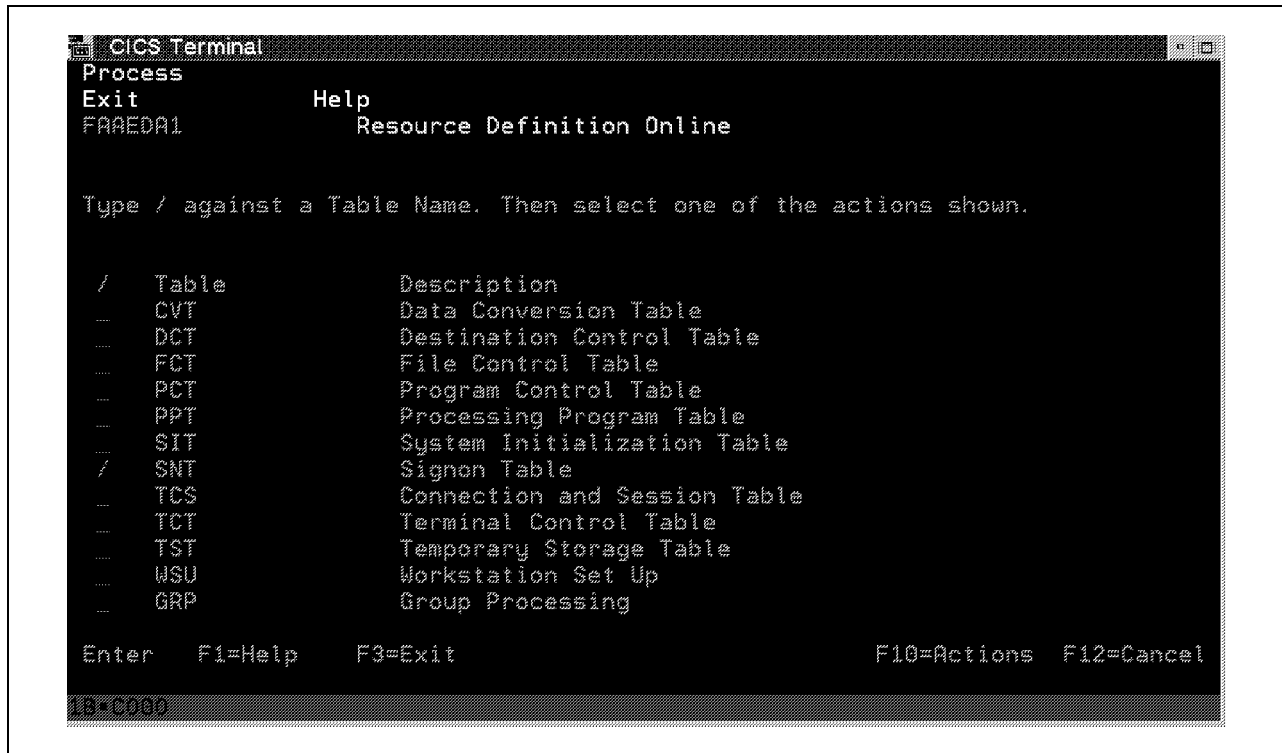


Figure 131. CICS Terminal Window for Resource Definition Online

7. To add users, press the F10 key on your keyboard to for activating the action bar on the Signon Table (Figure 132 on page 76). Move the cursor to the *Add* function by using the tab key and press the Ctrl key again.

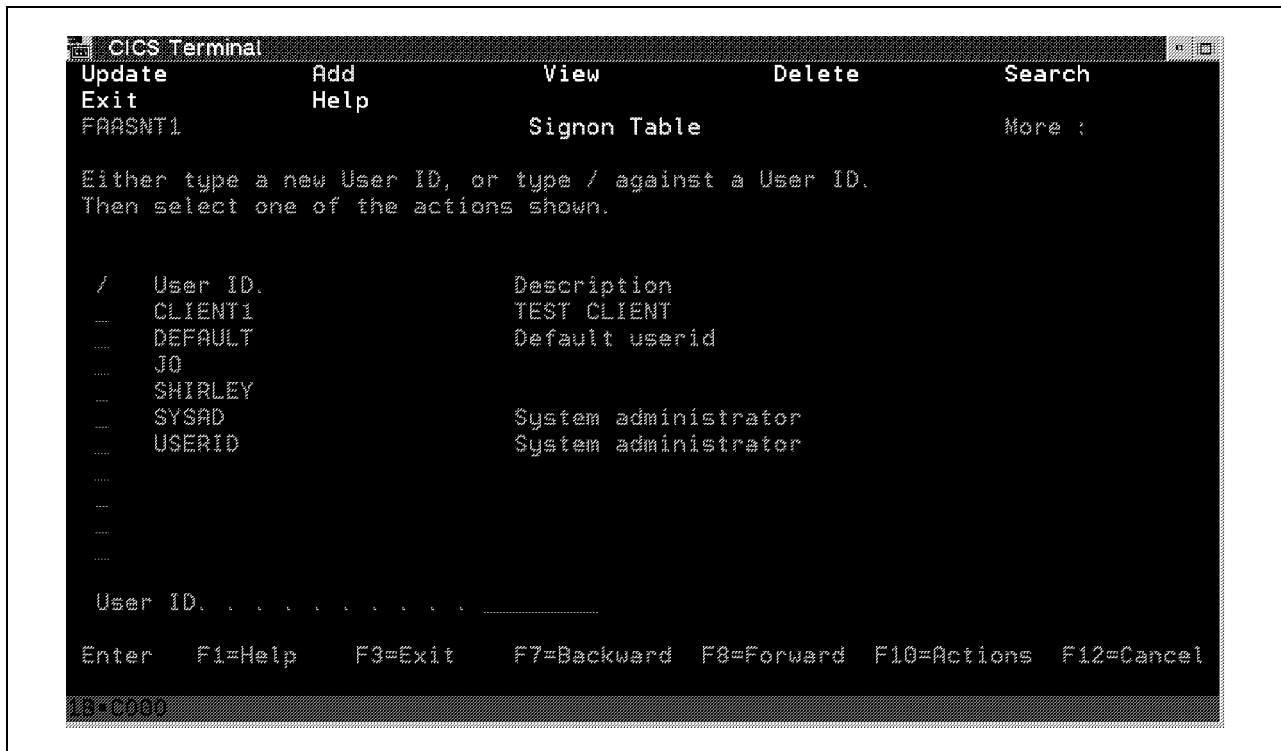


Figure 132. CICS Terminal Window for Signon Table

8. You can add users on Signon Table-1 (Figure 133 on page 77). Type a new user ID in the *User ID* entry field and the password in the *Password* field.

An *authorization to use tables* is required. The default is N. This specifies whether or not the user has access to the control tables.

The *operator class* and *operator keys* fields are provided for applications compatibility with host programs. Both may be used by the ASSIGN command. Leave them blank because none of the host programs in this context require it.

Press the F10 key again to activate the action bar. Use the tab key to reach the **Add** command and press the Ctrl key.



Figure 133. CICS Terminal Window for Signon Table-1

9. The Signon Table-1 shows this message in the command line:

Record added successfully. Press Enter to continue

Follow the request and press the Ctrl key again.

10. The Signon Table-1 appears again, allowing you to add more user IDs.

Press the F3 key. You have finished establishing user IDs and need not work with the administrator ID any longer.

If you want to run the first client/server application to see how it works, follow the instructions in 2.3.1.

2.3 Running CICS Client/Server Applications

After you have learned how to install and handle the CICS Client/Server environment, you may want to run your first application. Before you begin, read the following information for a better understanding of your environment.

Explanation

Libraries and Objects

When your COBOL program links libraries (*.LIB) or object files (*.OBJ), then you must specify the path where the object files are located. Declare these paths using the **LIB** command in the COBOL tools setup.

Copybooks

When using copybooks located in other directories, specify the paths of these copybooks using the **SYSLIB** command in the COBOL tools setup.

Dynamic Link Library

After successful compilation of a subprogram, the system creates a dynamic link library (DLL) file. This file resides in the path where the project is located. Before you can run your program, copy this DLL file into a directory that is specified in the CICSENV.CMD file in the CICSWRK environment variable. This variable is set to the \CICS300\RUNTIME directory but you can add your own path by specifying a value for UserWrk in the CICSENV.CMD file. You find the CICSENV.CMD file in the path D:\CICS300\RUNTIME where D is the drive in which CICS is installed.

SQLLIB for Precompile

If a COBOL program goes through the CICS precompiler before the COBOL compile starts, the compiler needs information on where the CICS precompiled program is located. This path has to be defined in the UserCobWork variable.

To specify the correct directory, open an OS/2 window and edit the CONFIG.SYS file by typing:

```
e C:\CONFIG.SYS
```

in the OS/2 window. Here, C is the drive where OS/2 is installed. Next, inspect to find where the TMP is located. This path is likely to be either D:\SQLLIB\TMP or D:\IBMCOBOL\TMP. Specify this path as, for example,

```
UserCobWork = 'D:\SQLLIB\TMP'
```

in the CICSENV.CMD file. This path is supposed to be set as the TMP OS/2 environment variable.

2.3.1 Running the First CICS Client/Server Application

Sample Project 5 of the VisualAge for COBOL samples is an application with a client intended to run on one OS/2 system and a server intended to run on another. The application uses CICS for communicating between the client and the server and uses online data instead of accessing DB2.

Sample Project 5 is appropriate to be our first client/server application.

The Sample 5 application works as follows:

The application of the server is called by the client using the ECI of CICS for OS/2. The server application returns a list of employees with associated data based on the client's request.

The client application is divided in two parts. The GUI program edits input from the GUI screen, calls the service calculation and the search subroutine, and fills the GUI screen. It uses ECI work-fields that contain the information needed by the CICS ECI facility for invoking the server program. And it uses an ECI-commarea for the CICS communication between the OS/2 client and the OS/2 server. The service calculation subroutine includes special program logic. It accepts a date, calculates, and returns the service length.

Running the program requires you to do different work on the server than on the client.

The server application machine must have IBM VisualAge for COBOL for OS/2 and CICS for OS/2 installed.

The client application machine needs only IBM VisualAge for COBOL for OS/2 installed.

2.3.1.1 Handling the Application on the CICS Server

Do the following on the CICS server:

1. Open the **VisualAge COBOL** icon, then the **Samples** icon included in the VisualAge COBOL icon view. Next, open the **Sample 5** icon (included in the Samples icon view), and finally open the **Server** icon. The Server-Icon view window (Figure 134) then appears.

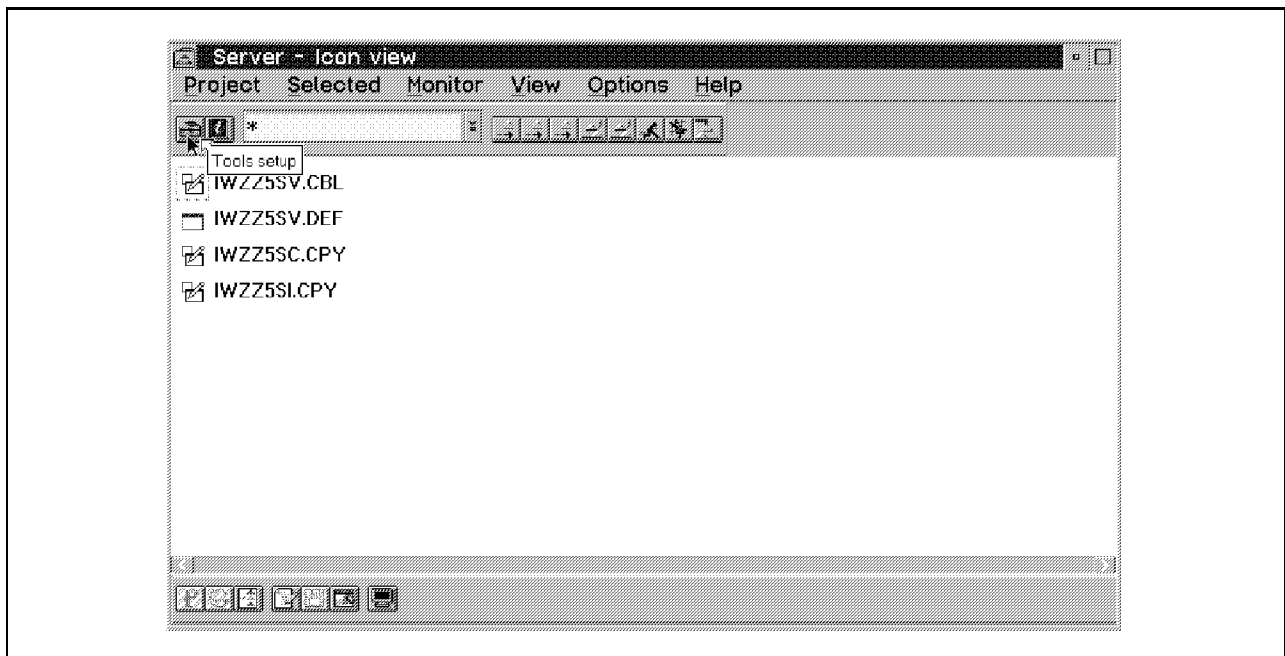


Figure 134. Server - Icon View Window for the CICS Application

2. Open the **Compiler** options, scroll to the **Link** options (Figure 135 on page 80), and ensure that the module definition file IWZZ5SV.DEF is set.

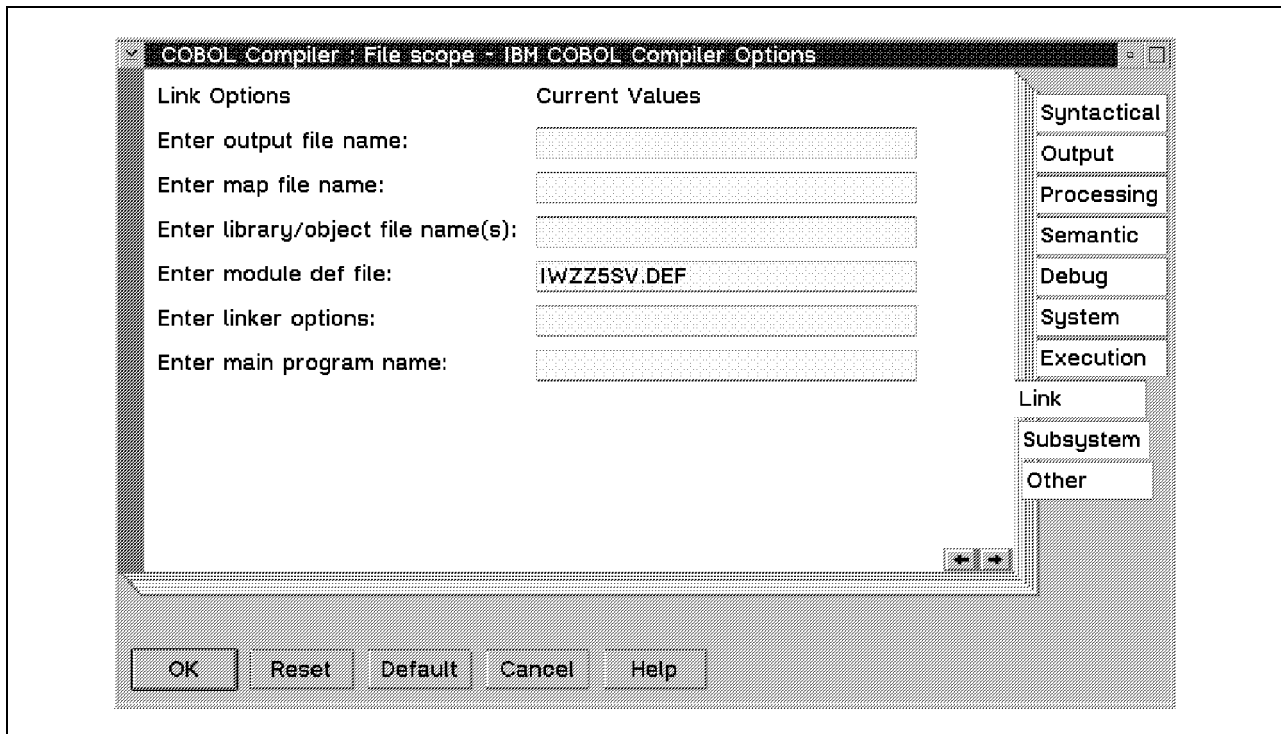


Figure 135. COBOL Compiler: File scope-IBM COBOL Compiler Options Window—Link Options

3. Scroll to the **Subsystem** tab on the next page (Figure 136) and make sure that **Preprocess for CICS** is selected.

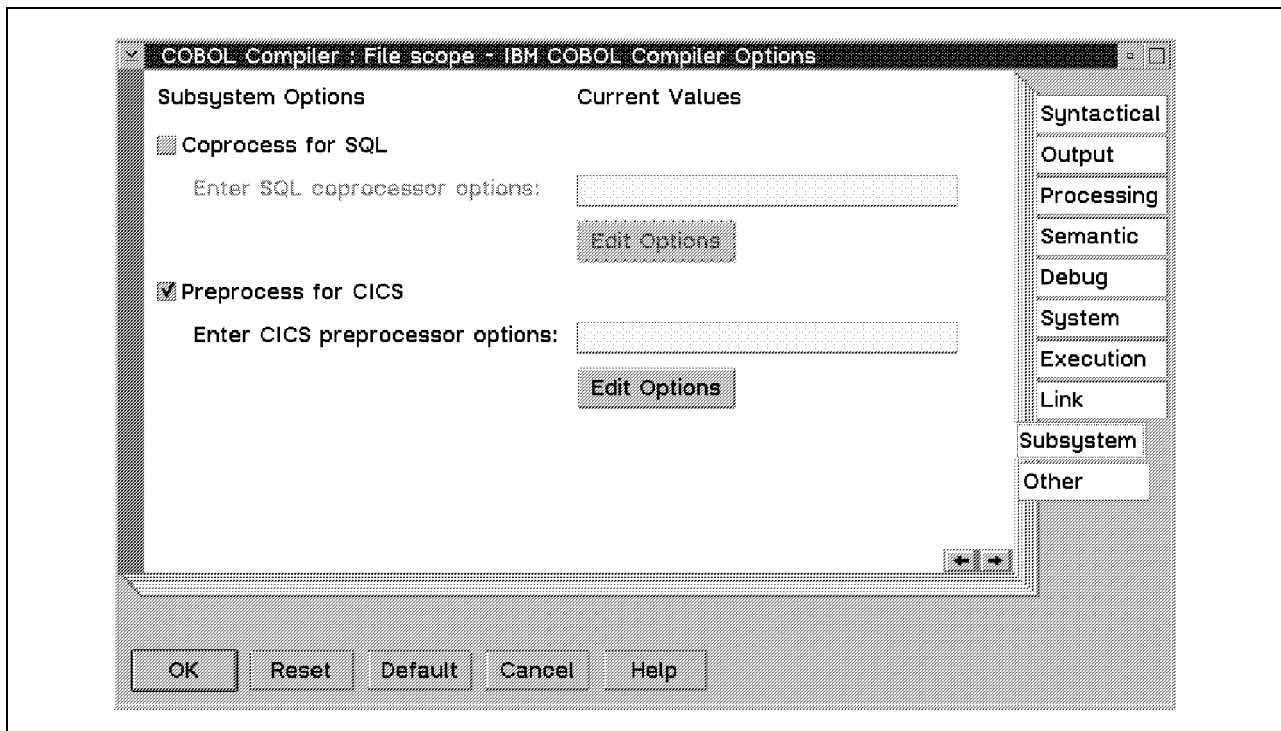


Figure 136. COBOL Compiler: File scope-IBM COBOL Compiler Options Window—Subsystem Options

Click on **OK** to close the **compiler** options.

4. Select the **Build** command in the **Project** menu bar to start the build procedure.

When the build procedure is finished, the monitor in the Server-Icon view window will show a successful completion ion message.

Dynamic Link Library

Before you can run your program you must copy the IWZZ5SV.DLL file in a directory that is specified in the CICSENV.CMD file in the UserWrk variable. In our configuration, the correct path for this DLL file is D:\IBMCOBOL\DLL.

5. Start the CICS by double-clicking on the appropriate icon on the IBM CICS for OS/2 folder. It is not necessary to log on. When the CICS Terminal V123 appears, the start process of CICS is finished.

2.3.1.2 Handling the Application on the CICS Client

Complete the following steps on the client machine to run the program:

1. On the Desktop, double-click on the **VisualAge COBOL** icon. Open the **Samples** icon and double-click on the **Sample 5** icon. The Sample Project 5-Icon View window appears (Figure 137).

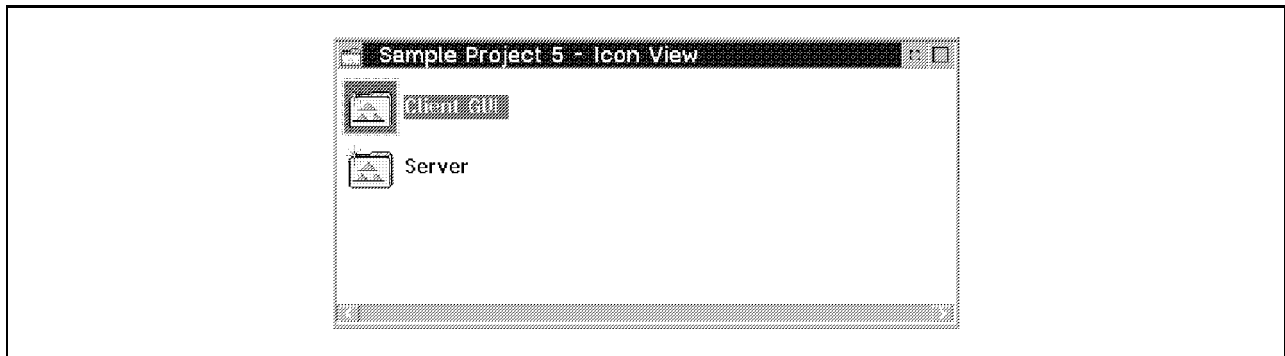


Figure 137. Sample Project 5—Icon View Window

2. Double-click on the **Client GUI** icon. The Client GUI-Icon view window appears (Figure 138 on page 82).

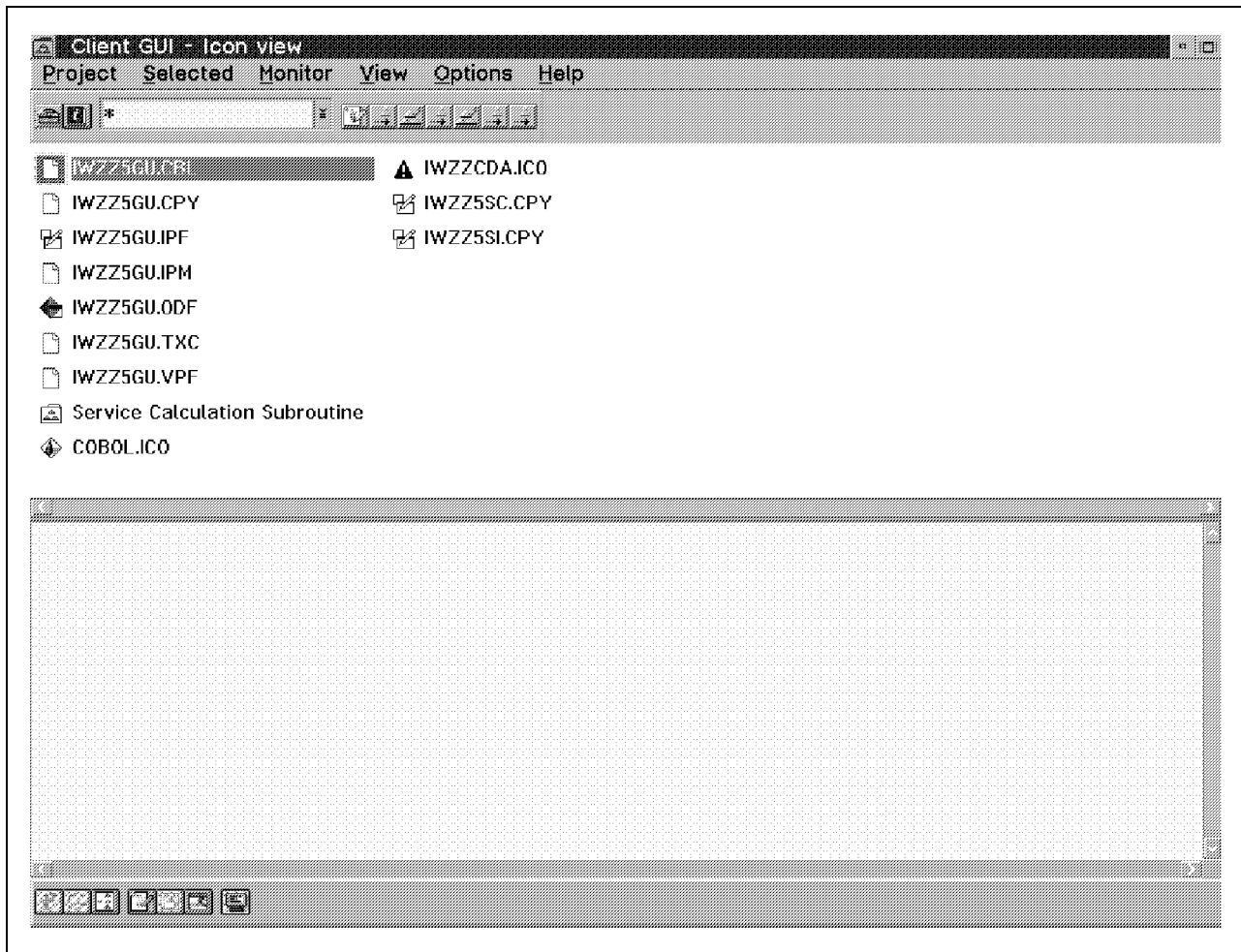


Figure 138. Client GUI-Icon View Window

3. Compile the program by using the build command. To do so, click on **Project** on the menu bar and select **Build** from the pull-down menu. The build procedure starts. After the successful compilation, as shown in Figure 139 on page 83, the client is able to access the server part of the application.

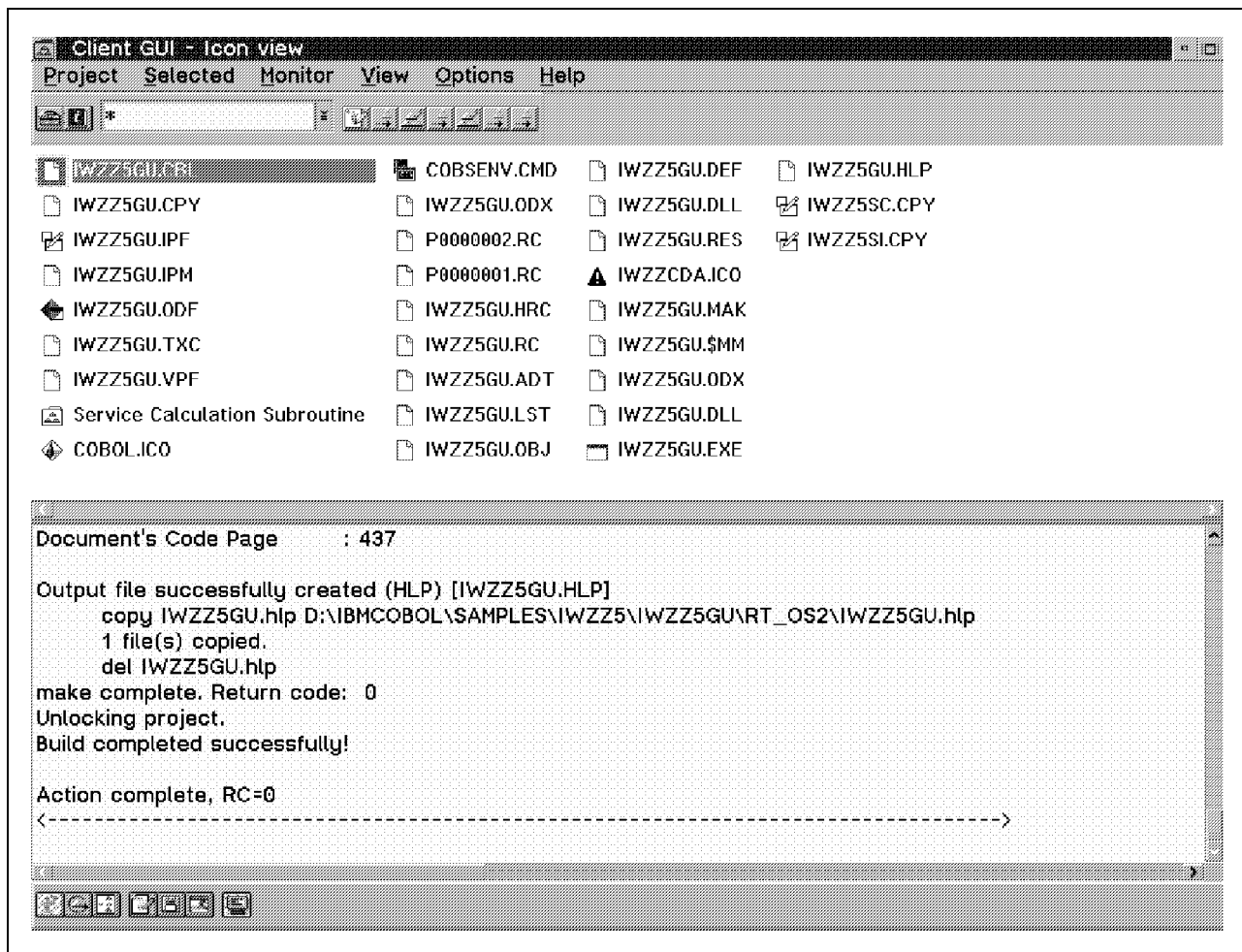


Figure 139. Client GUI-Icon view Window for Compilation

4. Start the CICS Client by double-clicking on the appropriate icon in the CICS folder. Ensure that CICS is also started on the server machine.
5. To run the application on the client, select **Project** on the menu bar and click on **Run** from the pull-down menu.

The Employee Lookup window appears (Figure 140 on page 84). Activate the ECI—Call to the server by clicking on **Search** in this window. The server program includes the online data and calls the subroutine that calculates a number using these online data. The subroutine supplies the calculated number to the server program. The server program returns it to the client program together with the online data. The client program then displays all the received data in the GUI window.

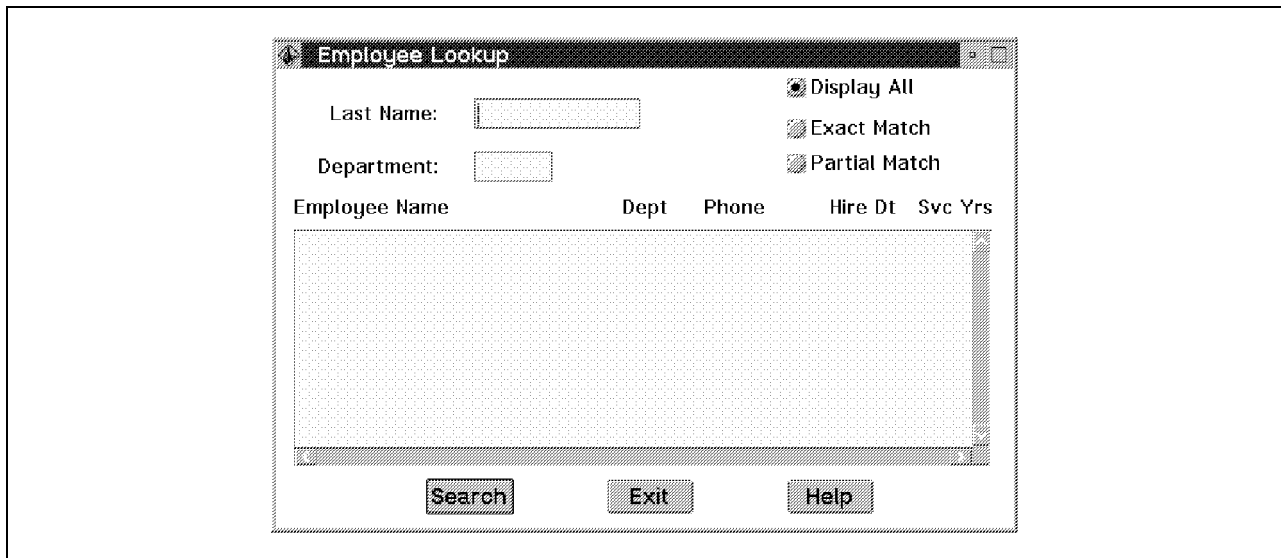


Figure 140. Employee Lookup Window

2.3.2 Running a CICS Client/Server Application with Database Access

Now extend the program by using DB2 data (instead of online data). You can use the Employee Table that is included in the Samples database in DB2. With the LAN Requester already installed on the CICS server machine, you can access data remotely from the LAN Server. In this way, you have combined CICS transactions and you access DB2 using LAN in one program.

All necessary files are stored on the diskette included with in this book. The project name is EMPLLU, for Employee Lookup.

The best way to see how CICS works is to install different parts of the application on different physical machines in the environment.

The client part is assigned to the execution machine where VisualAge for COBOL must be installed. When you use the previously installed environment, then on this machine the CICS Client is already installed. However, this product has not necessarily been installed for this application, because the client part will not call the CICS precompiler.

The server part must be installed on the machine that includes the CICS Server, because CICS is needed to precompile the program. Further, VisualAge for COBOL is needed for the installation and generation of the project and LAN requester is needed to access DB2 server. Access the database located on the third machine, where the DB2 Server and the LAN Server are installed.

First, install and configure the server part of the application, followed by the client part. Finally, prepare the environment to compile the application so that it may run.

2.3.2.1 Installing the Server Part of the Application

First install the server part on the server machine, which contains the CICS Server and is able to access DB2 on the DB2 Server using LAN. Do the following:

1. Open the VisualAge COBOL icon view by double-clicking on the appropriate icon on the OS/2 Desktop. The VisualAge COBOL-Icon View window appears. Open the Templates icon view by double-clicking on the **Templates** icon and the Templates-Icon View window appears (Figure 141). Move the mouse pointer to the **COBOL Project** icon using mouse button 2 and move the icon to the empty space on the Desktop. Release mouse button 2 and the **COBOL Project** icon is displayed on the Desktop.

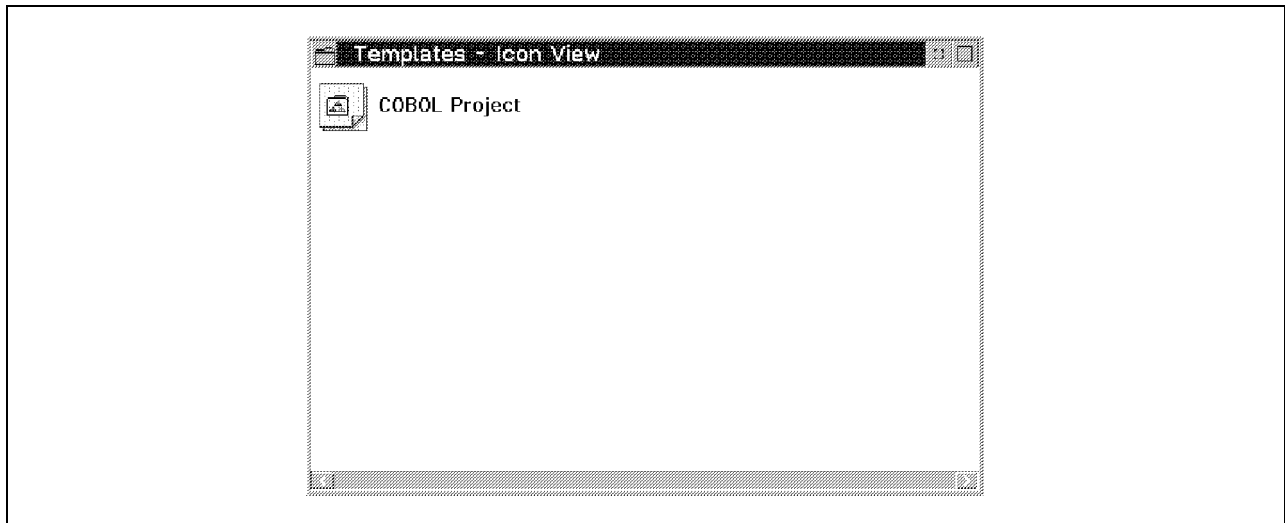


Figure 141. Templates-Icon View Window

2. Move your mouse pointer to the **COBOL Project** icon on the Desktop and click mouse button 2. A pop-up menu appears. Select **Settings** from the pop-up menu. The **Target** tab of the COBOL Project-Settings window is displayed (Figure 142 on page 86). Type `EMPLPLUSV.DLL` in the *Name* field.

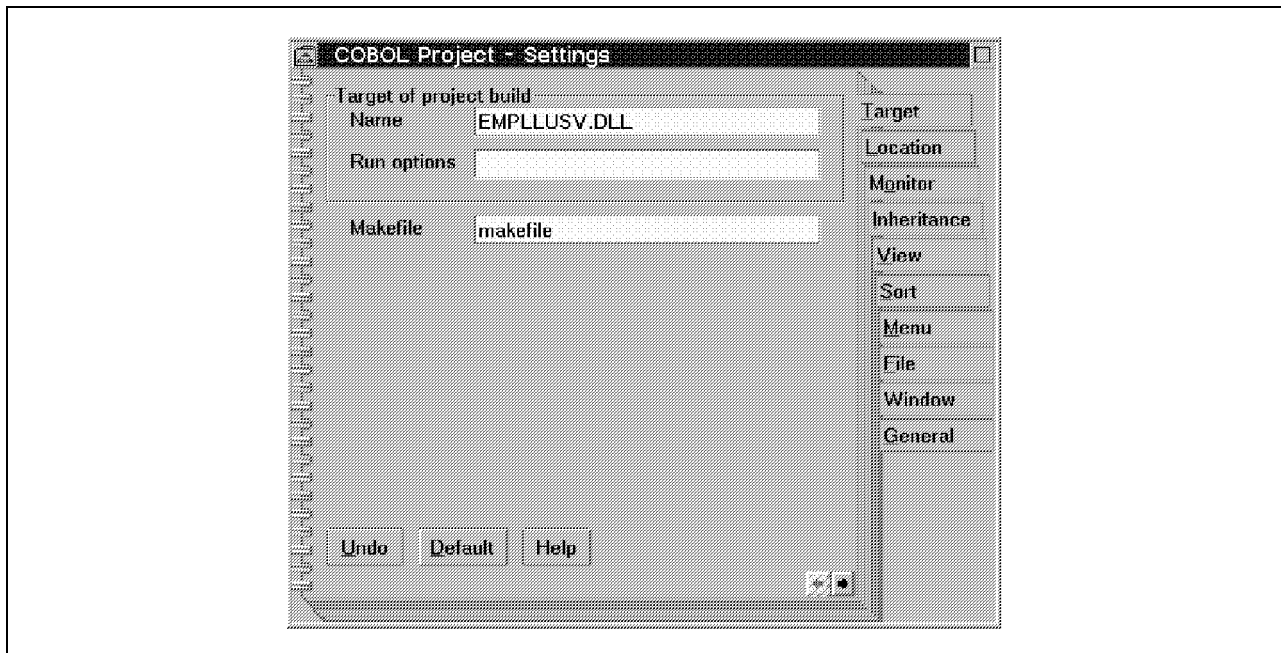


Figure 142. COBOL Project-Settings Window for the Target (Server)

3. Click the **Location** tab to scroll to the next page of the COBOL Project-Settings window. (Figure 143). Type D:\IBMCOBOL\EMPLLU and D:\IBMCOBOL\EMPLLU\SERVER in the *Source directories for project files* field, where D should be the drive on which you have installed the IBM VisualAge for COBOL for OS/2. Click on the *Working directory* entry field.

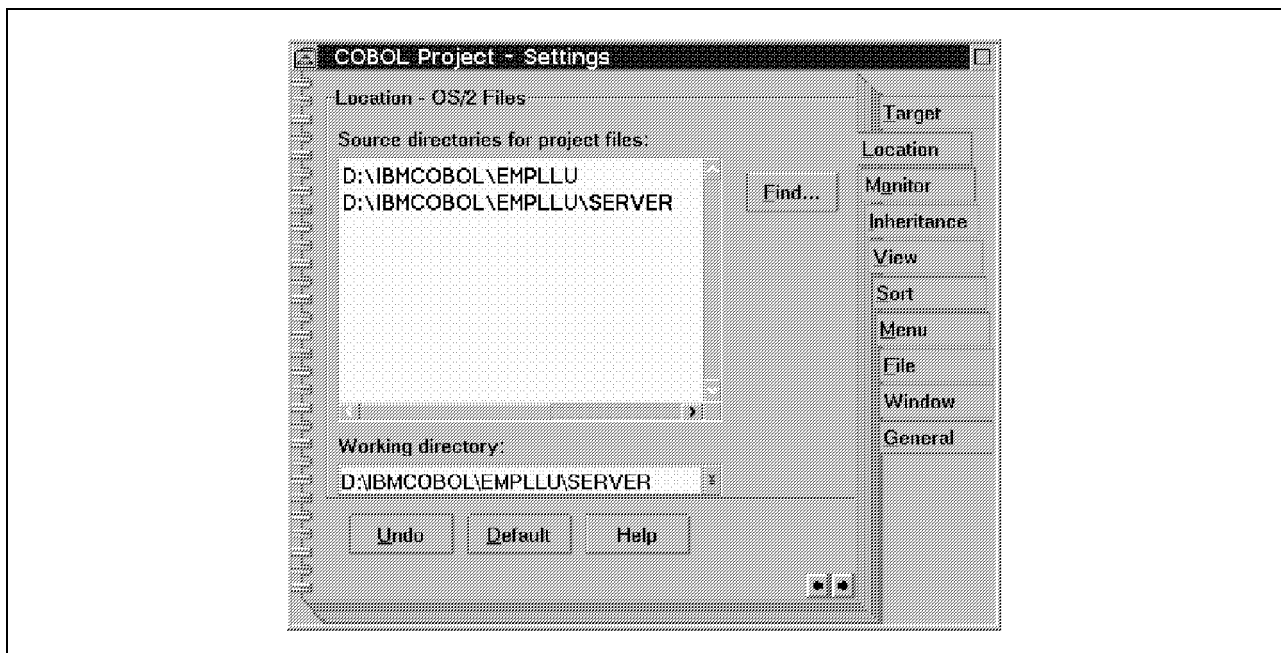


Figure 143. COBOL Project-Settings window for the Server Location

4. The Create directories window then appears. (Figure 144 on page 87). Click on **Yes** to create the directories on your workstation.

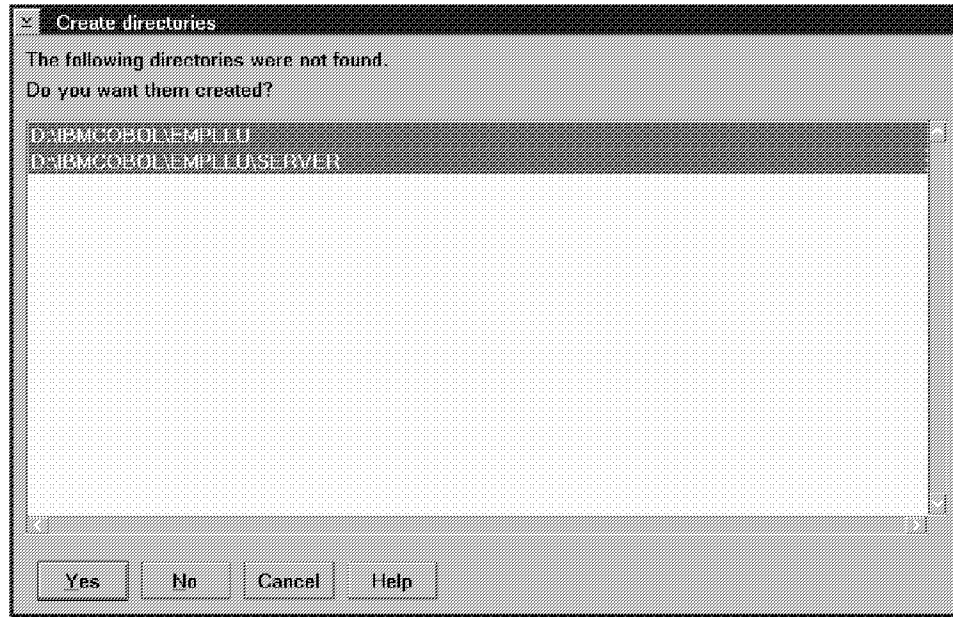


Figure 144. Create directories Window for the Server

5. The COBOL Project-Settings window for Location is shown again. Select `D:\IBMCOBOL\EMPLLU\SERVER` from the *Working directory* field. Scroll to the **General** page and type `EMPLLU Server` into the *Title* field (Figure 145).

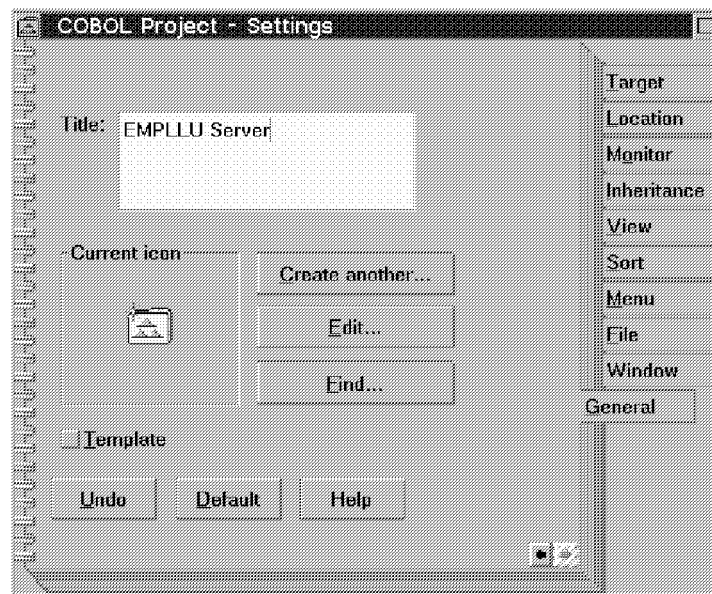


Figure 145. COBOL Project-Settings Window for Server (General)

6. Scroll back to the **File** page. Click on *Name* entry field: the default name for the physical name then changes automatically to the name given by the system (Figure 146 on page 88).

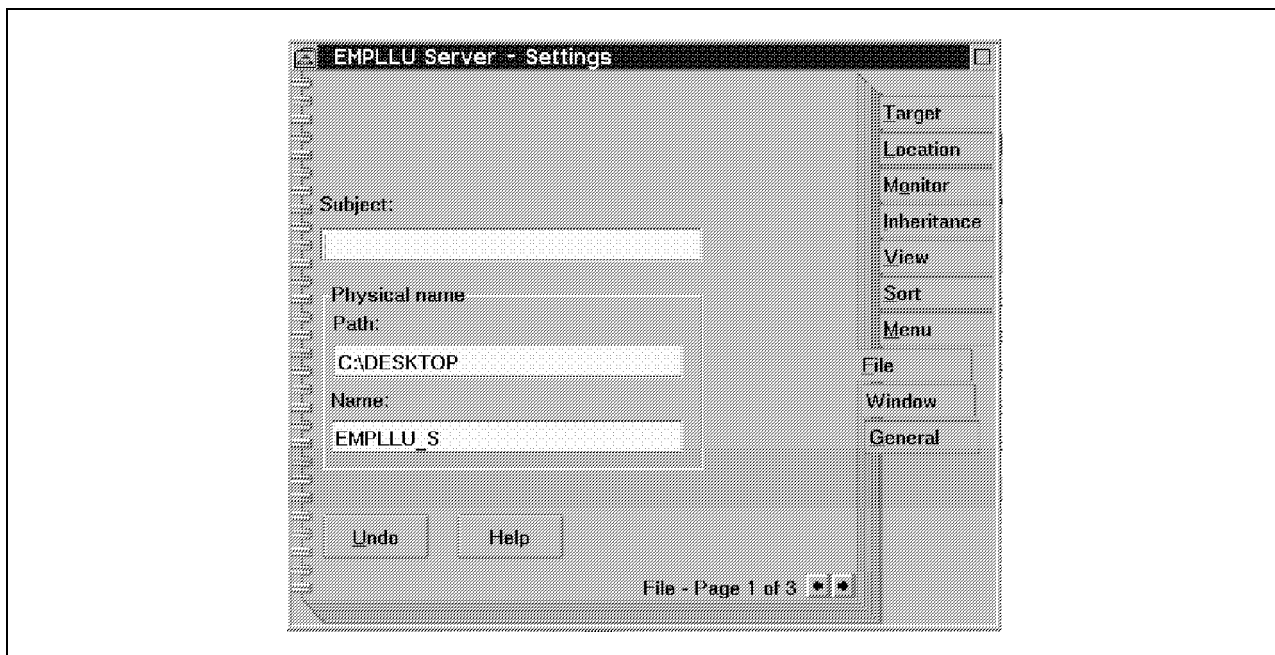


Figure 146. EMPLLU Server-Settings Window

7. The settings for the server program are now complete.

Next, create the project for the service calculation routine. Create a second project icon on the Desktop as described on page 80. Open the settings of this icon. The COBOL Project - Settings window appears with the **Target** page. Accept the defaults on this page and select the **Location** tab (Figure 147). Type `D:\IBMCOBOL\EMPLLU\SERVICE` in the *Source directories for project files* field, click on the *Working directory*, field to ensure the creation of this directory, and select `D:\IBMCOBOL\EMPLLU\SERVICE` as the Working directory.

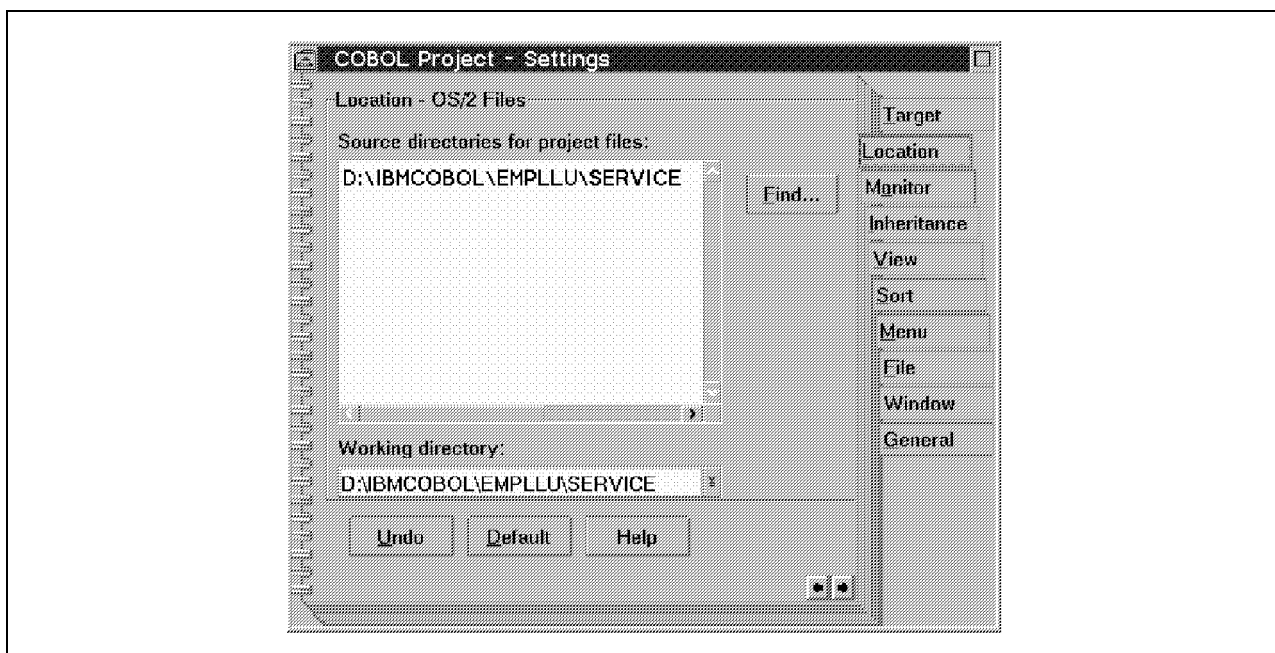


Figure 147. COBOL Project-Settings Window for the Target Location

8. Scroll to **General** page and type EMPLLU Service Calculation into the *Title* field (Figure 148 on page 89).

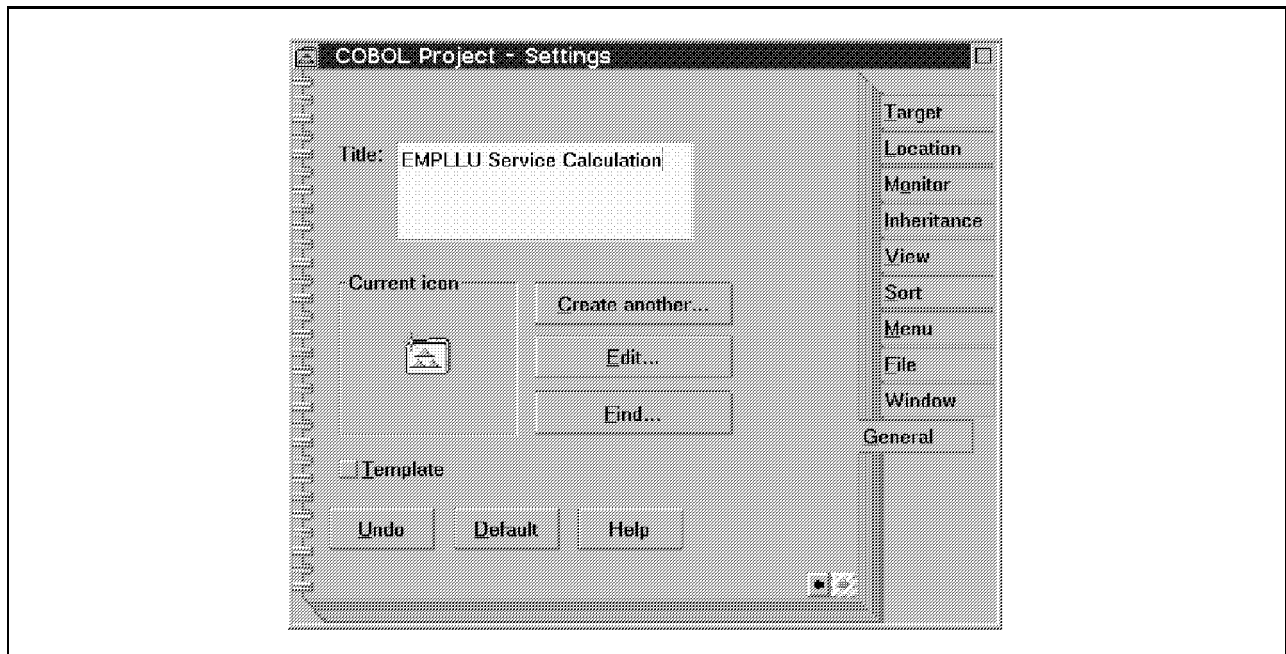


Figure 148. COBOL Project-Settings Window for General Settings

9. Scroll back to the File page. (Figure 138 on page 83). Click on the *Name* entry field. The default path for the physical name then changes to the name the system has given it (Figure 149).

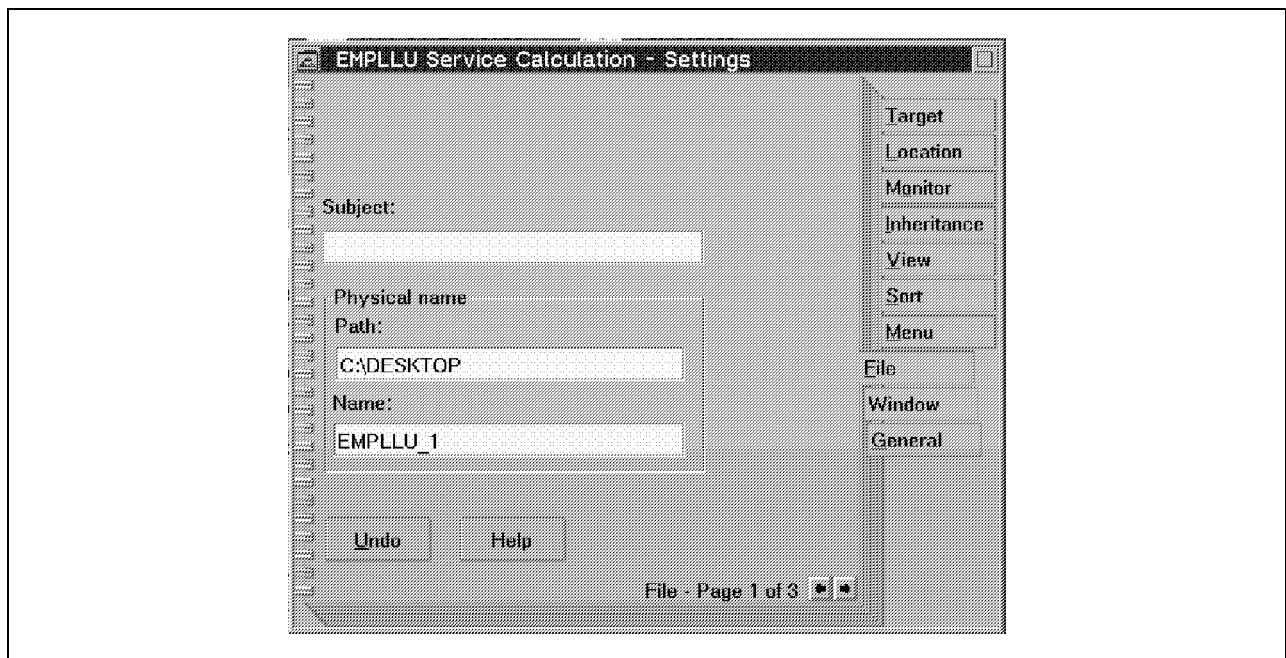


Figure 149. EMPLLU Service Calculation-Settings Window

The settings of the service calculation routine are complete.

10. For copying the files from the diskette into the appropriate directories on the hard disk, place the diskette into the A drive, open an OS/2 window, and type the following commands in this window:

```
COPY A:\EMPLLU\SERVER D:\IBMCOBOL\EMPLLU\SERVER
COPY A:\EMPLLU D:\IBMCOBOL\EMPLLU
COPY A:\EMPLLU\SERVICE D:\IBMCOBOL\EMPLLU\SERVICE
```

where D is the drive in which you installed IBM VisualAge for COBOL for OS/2. Press Enter after each command. The system copies the files from the diskette in the specified paths.

11. Open the EMPLLU Server project by double-clicking on the **EMPLLU Server** icon on the Desktop. the EMPLLU Server—Icon view window appears (Figure 150).

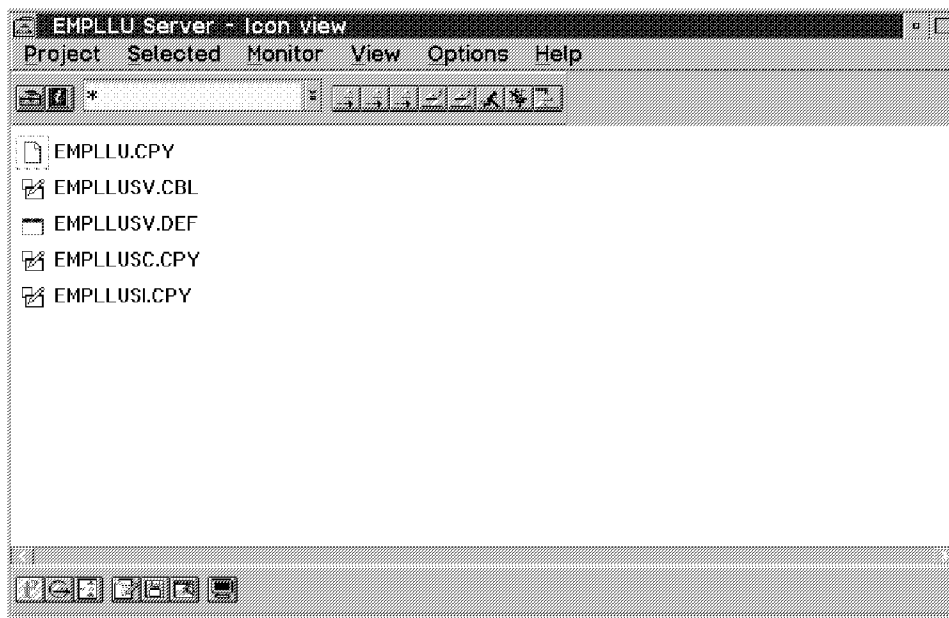


Figure 150. EMPLLU Server-Icon View Window

12. Click on the **Tools setup** icon, the leftmost icon on the icon bar in the EMPLLU Server—Icon view window. The EMPLLU Server-Tools setup window appears (Figure 151 on page 91).

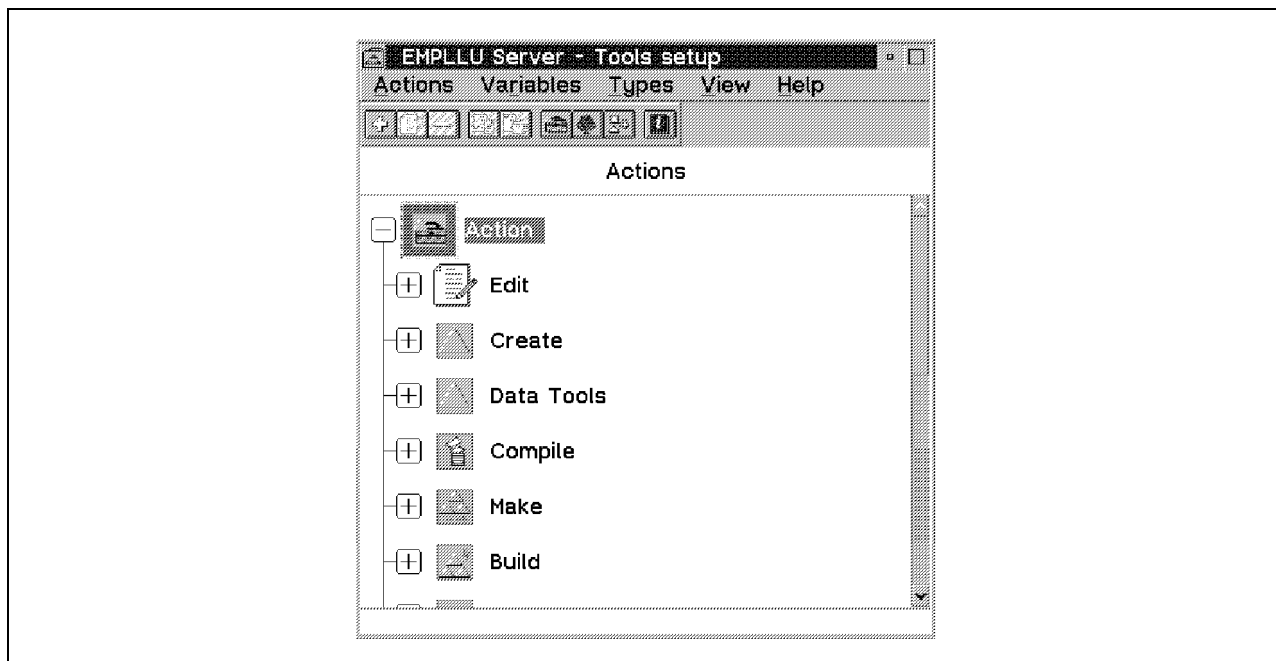


Figure 151. EMPLLU Server-Tools Setup Window

13. Select **Variables** from the menu bar and then select **Add...** from the pull-down menu to add environment variables. The Add Environment Variable window appears (Figure 152). Select **LIB** in the combination box and enlarge the String path into:

D:\IBMCOBOL\LIB;C:\MUGLIB;D:\SQLLIB\LIB;D:\IBMCOBOL\EMPLLU;
D:\IBMCOBOL\EMPLLU\SERVER;D:\IBMCOBOL\EMPLLU\SERVICE

where you have to ensure that you specify the correct drives. Click on **Add**. The variable is added to the environment variables of this project.

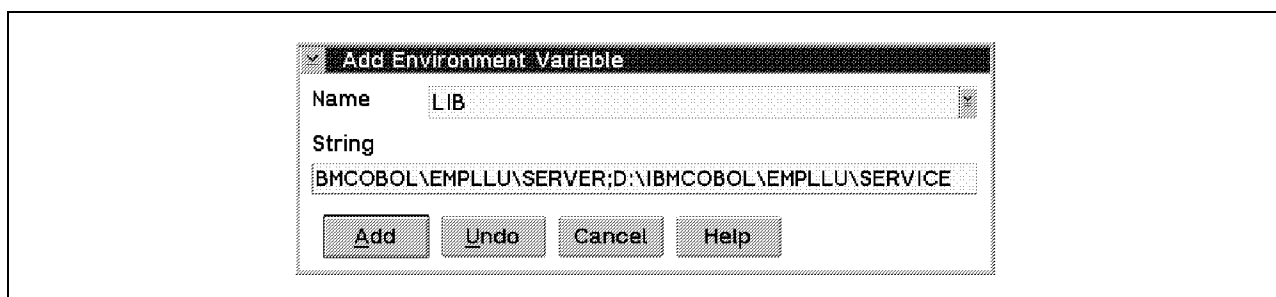


Figure 152. Add Environment Variable Window-LIB

14. In the same way, add the SYSLIB variable to the environment variables. To do this, select **Add...** from the **Variable** pull-down menu in the EMPLLU Server-Tools setup window. The Add Environment Variable window appears (Figure 152). You will not find SYSLIB in the combination box, so you must type it in the *Name* field. Change the entry of the *String* field in D:\IBMCOBOL\EMPLLU (Figure 153 on page 92). Click on **Add**.

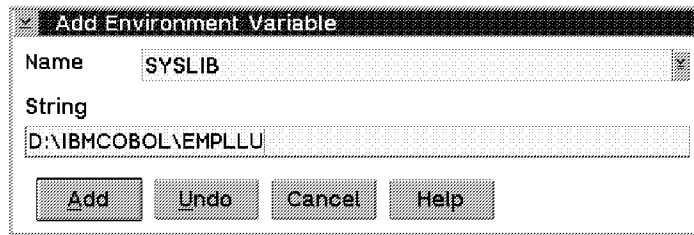


Figure 153. Add Environment Variable Window—SYSLIB

15. To close the EMPLLU Server-Tools setup window, double-click on the **System menu** icon located in the upper left-hand corner of the window.

Click on **Options** in the menu bar of the EMPLLU Server pull-down menu and select **Compile** to receive the COBOL Compiler: File scope-IBM COBOL Compiler Options window. The **Syntactical** tab is the first page of the compiler options (Figure 154). Activate **Process COPY, BASIS, and REPLACE statements**. The current value changes to LIB.

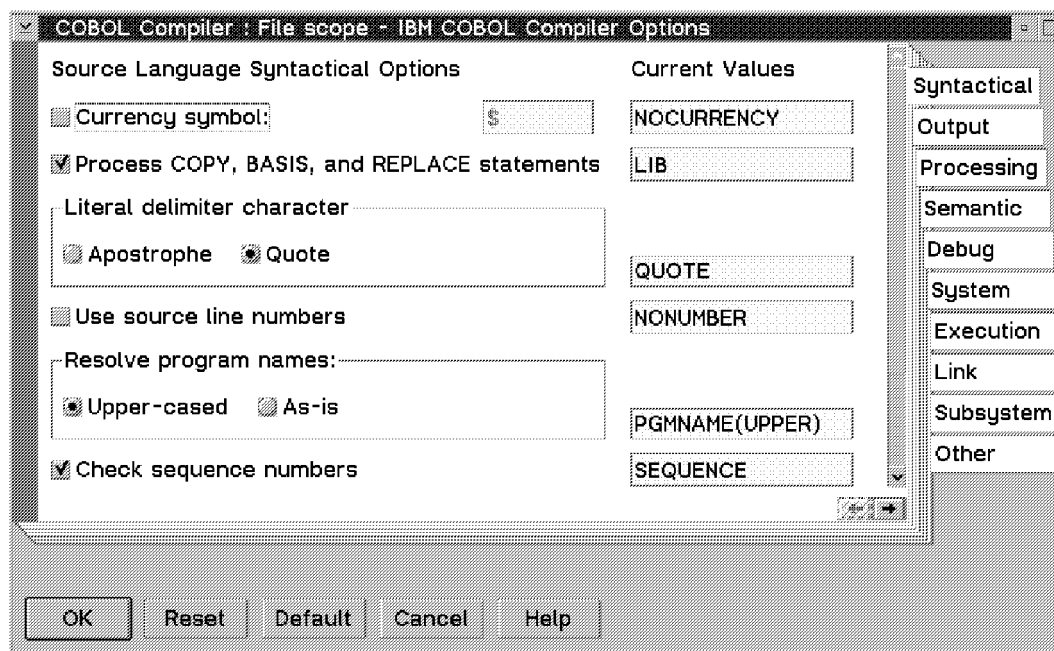


Figure 154. COBOL Compiler: File scope-IBM Compiler Options Window for Syntactical Options

Scroll to the **Link** options (Figure 155 on page 93) and type DB2API.LIB EMPLLUSC.OBJ in the *library/object file name* field and EMPLLUSV.DEF in the *module definition file* field.

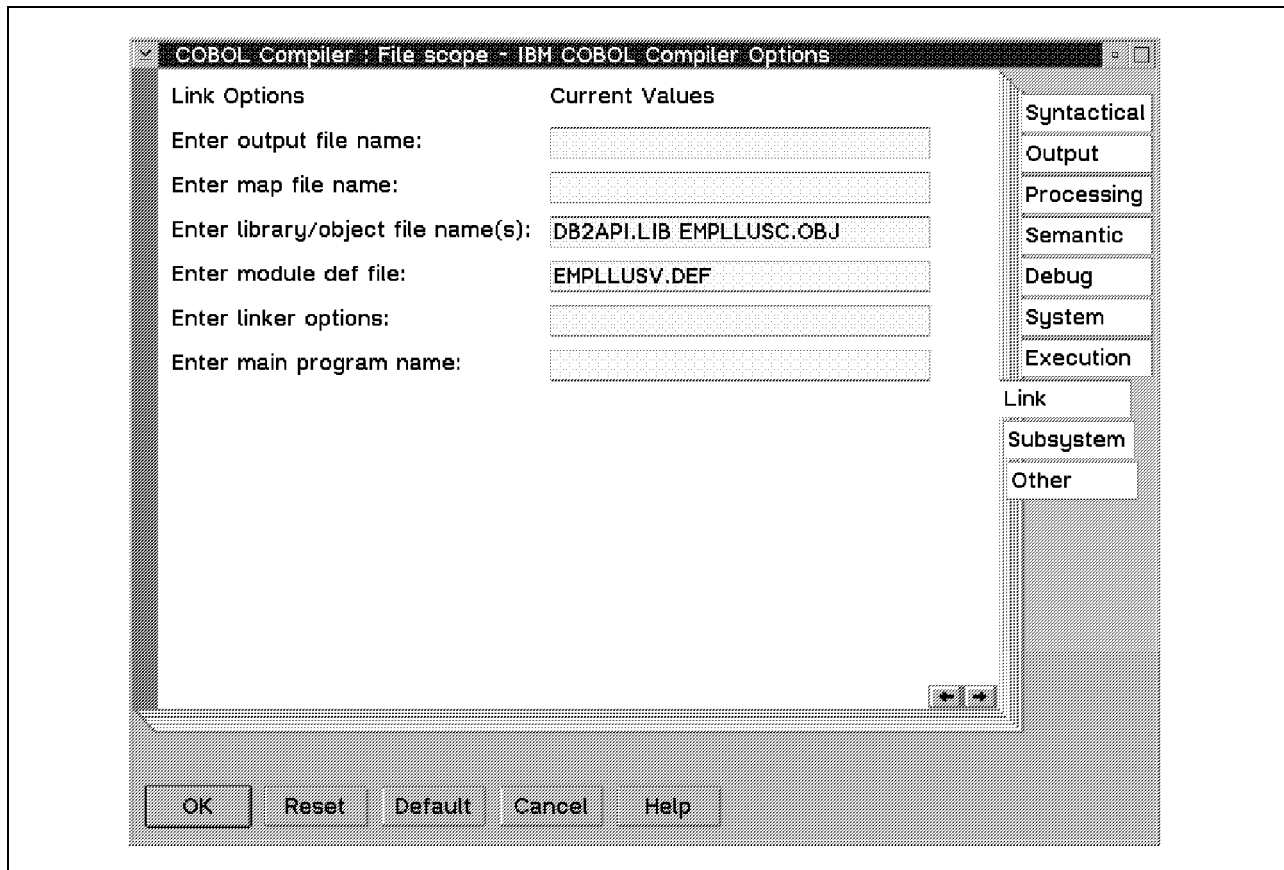


Figure 155. COBOL Compiler: File scope-IBM COBOL Compiler Options Window—Link Options

16. Scroll to the **Subsystem** options (Figure 156 on page 94). Select **Coprocessor for SQL**, click on **Edit Options**, and type

```
DATABASE SAMPLE BINDFILE PACKAGE
```

in the upcoming Edit SQL Coprocessor Options window (Figure 102 on page 55). Then select **Preprocess for CICS**. Click on **OK** to save and leave the compiler options.

The compiler options for the server routine are now set.

Note: A date as it is retrieved from DB2 can have different formats. On OS/2, it depends on the `COUNTRY` definition in the `CONFIG.SYS` file.

With the corresponding DB2 precompiler option, you can specify what date format you want your application program to get from DB2.

Because the Service Calculation routine of our sample EMPLLU application is calculating with the date it retrieves from DB2, it is important that it gets this date in the correct format.

The EMPLLUSC structure which passes the parameters to the Service Calculation routine is based on the USA date format (mm/dd/yyyy). Therefore if your OS/2 workstation is set up with a `COUNTRY` code other than 001 you have to specify `DATETIME USA` as an additional SQL coprocessor option.

The complete coprocessor options are in this case:

```
DATABASE SAMPLE BINDFILE PACKAGE DATETIME USA
```

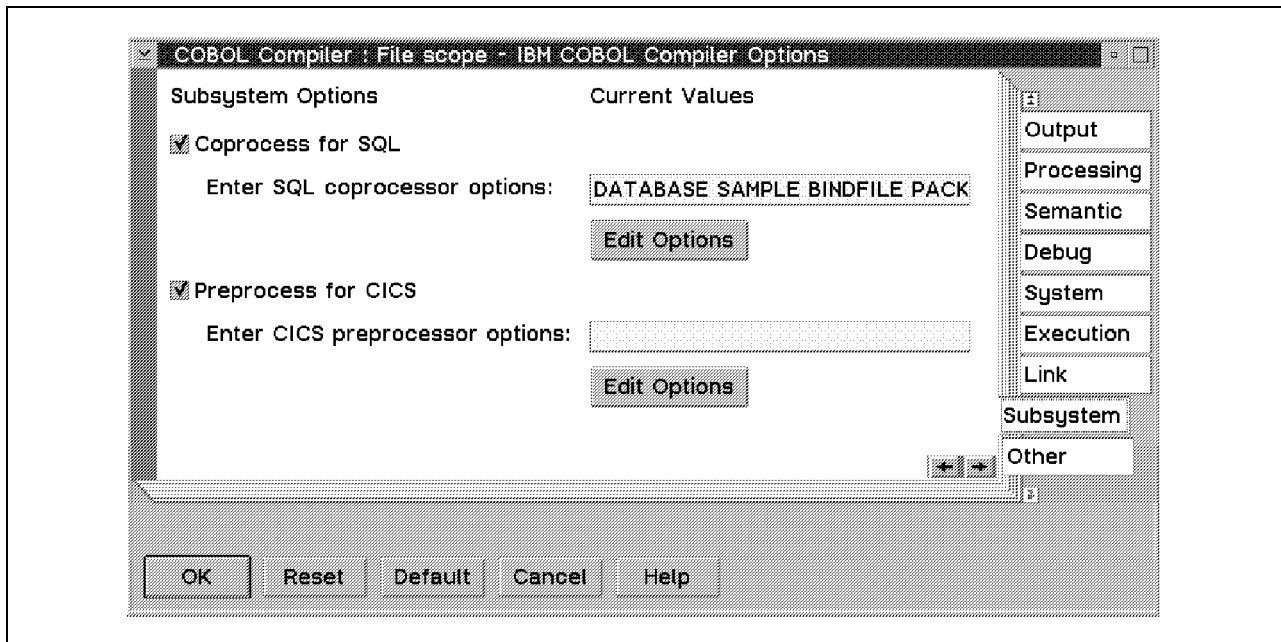


Figure 156. COBOL Compiler: File scope—IBM COBOL Compiler Options Window—Subsystem Options

17. Open the service calculation routine to specify a compiler option. Select the compiler options as described above.
18. On the **Syntactical** tab, click on **Process COPY, BASIS, and REPLACE statements** to activate this setting in the same way as you did with the compiler options for the server (Figure 154 on page 92). The current value changes to LIB.
19. Scroll to the **Other** options (Figure 157) and select **Compile programs but do not link**. Click on **OK** to save and leave the compiler options for the service calculation routine.

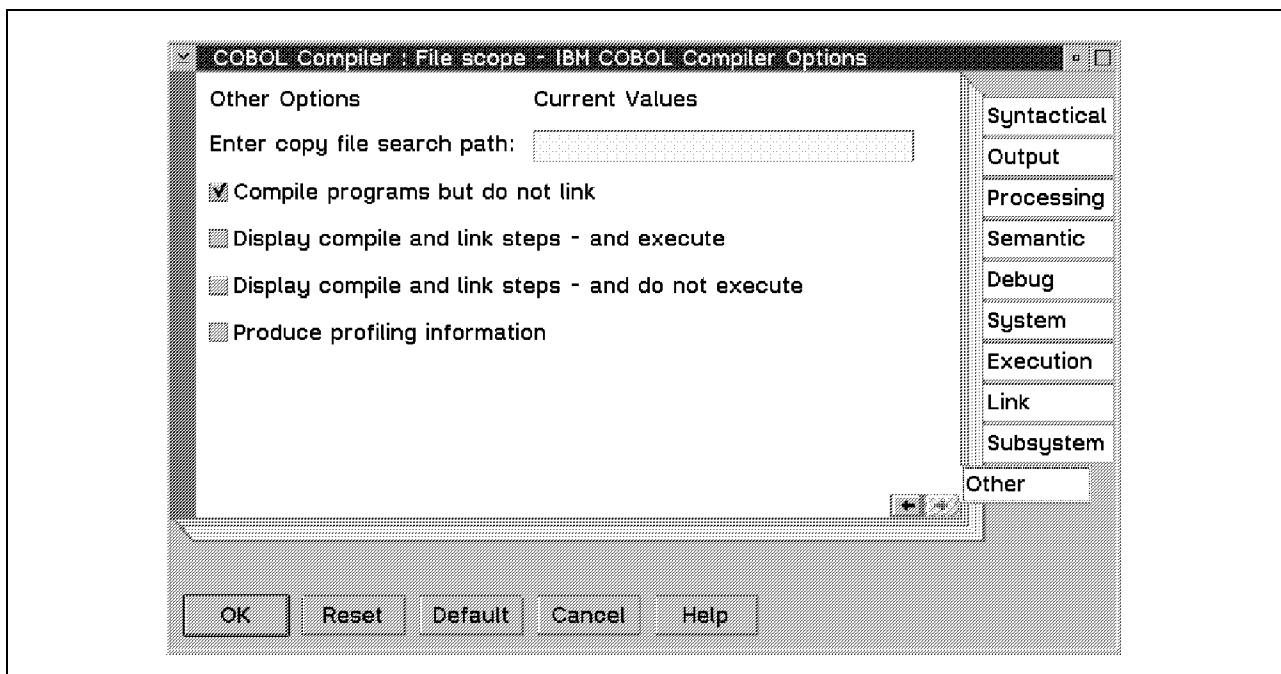


Figure 157. COBOL Compiler: File scope-IBM COBOL Compiler Options Window—Other Options

20. To close the service calculation-routine project, double-click on the **system menu** icon in the upper left-hand corner of the window.
21. Finally, copy the service calculation-routine project into the server project. To do this, move the mouse pointer to the **EMPLLU Service Calculation** icon on the Desktop. Using mouse button 2, move the icon over the **EMPLLU Server** icon on the Desktop and release mouse button 2.

The service calculation routine is now included in the server project; you need not compile it separately.

22. Make sure that DB2 for OS/2 Server is started on the server machine and that you have carried out the client setup as described in Section 1.2.3, "DB2 Client Setup" on page 37.
23. To compile the server project, open the project by double-clicking on the appropriate icon. Click on **Project** on the menu bar and select **Build**. After the successful build procedure, you may turn to the client machine.

2.3.2.2 Installing the Client Part of the Application

To install the client part of the application EMPLLU on the machine where the CICS Client is installed, do the following:

1. To create a new project, use the method described for the server. Do it again on this machine:

Open the VisualAge COBOL icon view by double-clicking on the appropriate icon on the OS/2 Desktop. The VisualAge COBOL-Icon View window appears. Open the Templates icon view by double-clicking on the **Templates** icon. When the Templates-Icon View window appears, move the mouse pointer to the **COBOL Project** icon. Using mouse button 2, move the icon to the empty space on the desktop. Release mouse button 2 and the **COBOL Project** icon is displayed on the desktop.

2. Move your mouse pointer to the **COBOL Project** icon on the desktop and click mouse button 2. A pop-up menu appears. Select **Settings** from the pop-up menu. The COBOL Project—Settings window for the project build target is displayed (Figure 158 on page 96). Type `EMPLLUGU.EXE` in the *Name* field and `EMPLLUGU.MAK` in the *Makefile* field.

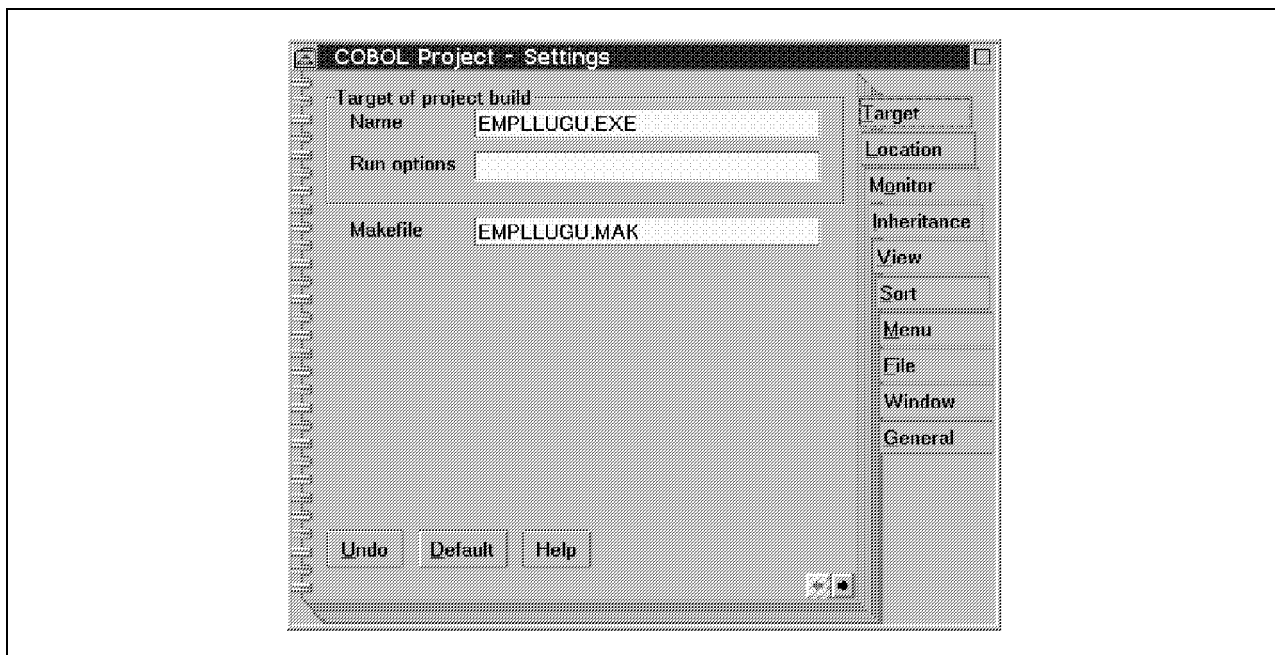


Figure 158. COBOL Project-Settings Window for the Target (Client)

3. Scroll to the next page of the COBOL Project-Settings (location settings) window (Figure 159). Type `D:\IBMCOBOL\EMPLLU` and `D:\IBMCOBOL\EMPLLU\RT_OS2` in the *Source directories for project files* field, where D should be the drive on which you have installed the IBM VisualAge for COBOL for OS/2. Ensure that `RT_OS2` is written in capital letters. Click on the *Working directory* field.

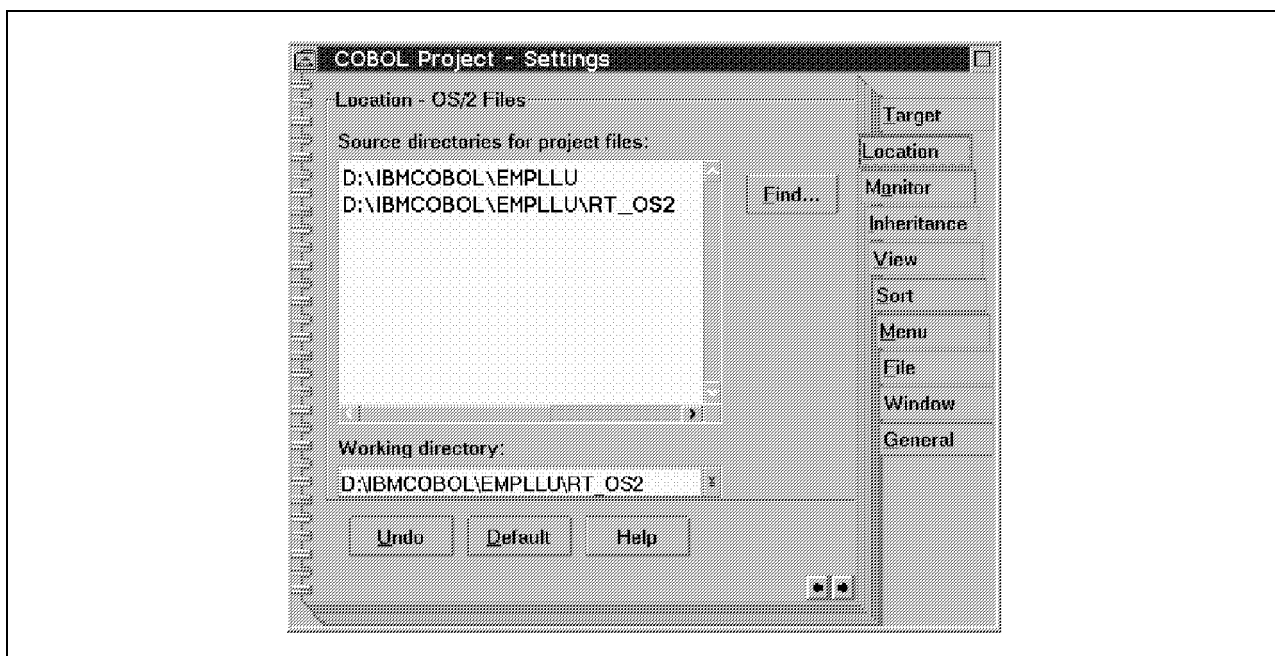


Figure 159. COBOL Project-Settings Window for Client Location Settings

4. The Create directories window appears. Click on **Yes** to create the directories on your workstation.

5. The COBOL Project-Settings Window for Location is shown again. Specify `D:\IBMCOBOL\EMPLLU\RT_OS2` in the *Working directory* field. Scroll to the Inheritance page (Figure 160 on page 97), select **COBOL Master Project** from the inheritance list, and click on **Remove**. The default for a templates project is COBOL Master Project, but you want to define a COBOL GUI Master Project. You have to define this inheritance for this project.



Figure 160. COBOL Project-Settings Window for Client Inheritance

6. To define a new inheritance, click on **Add...** on the window labeled Inheritance page of the COBOL Project-Settings. The window labeled Select a project to inherit from (Figure 161 on page 98) appears. In the **Directory** listbox, select the following paths:

- **C:**
- **Desktop**
- **VisualAge COBOL (or its abbreviation VISUALAG)**
- **Works**

In the *File* field, COBOL GUI Designer Master Project (or its abbreviation, VISUALAG) appears. Select it and click on **Inherit**.

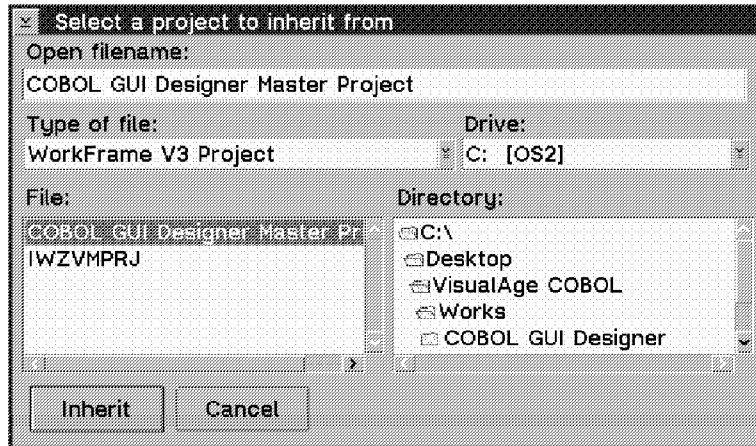


Figure 161. Select a project to inherit from Window

7. Scroll to the COBOL Project-Settings window general page and type `EMPLLU Client` in the *Title* field (Figure 162).

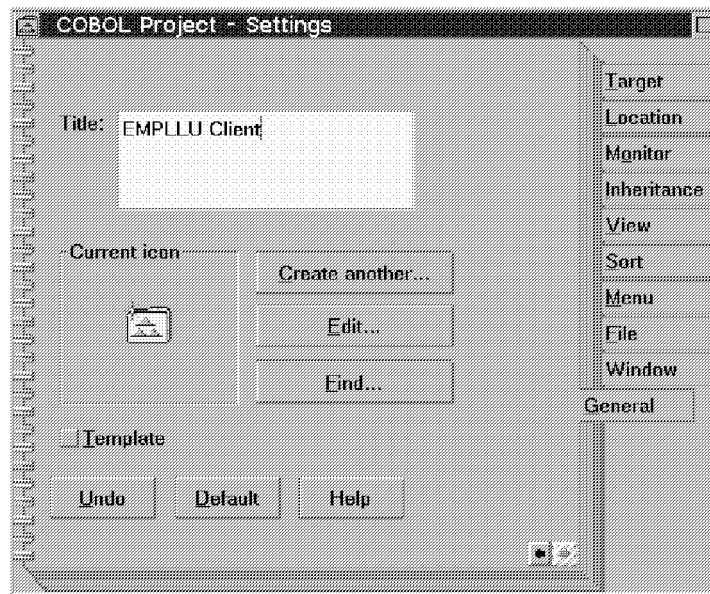


Figure 162. COBOL Project-Settings Window for General Client Settings

8. Scroll back to the File page (Figure 159 on page 96). Click on the *Name* entry field; the default path then changes to the physical name `EMPLLU Client` (Figure 163 on page 99).

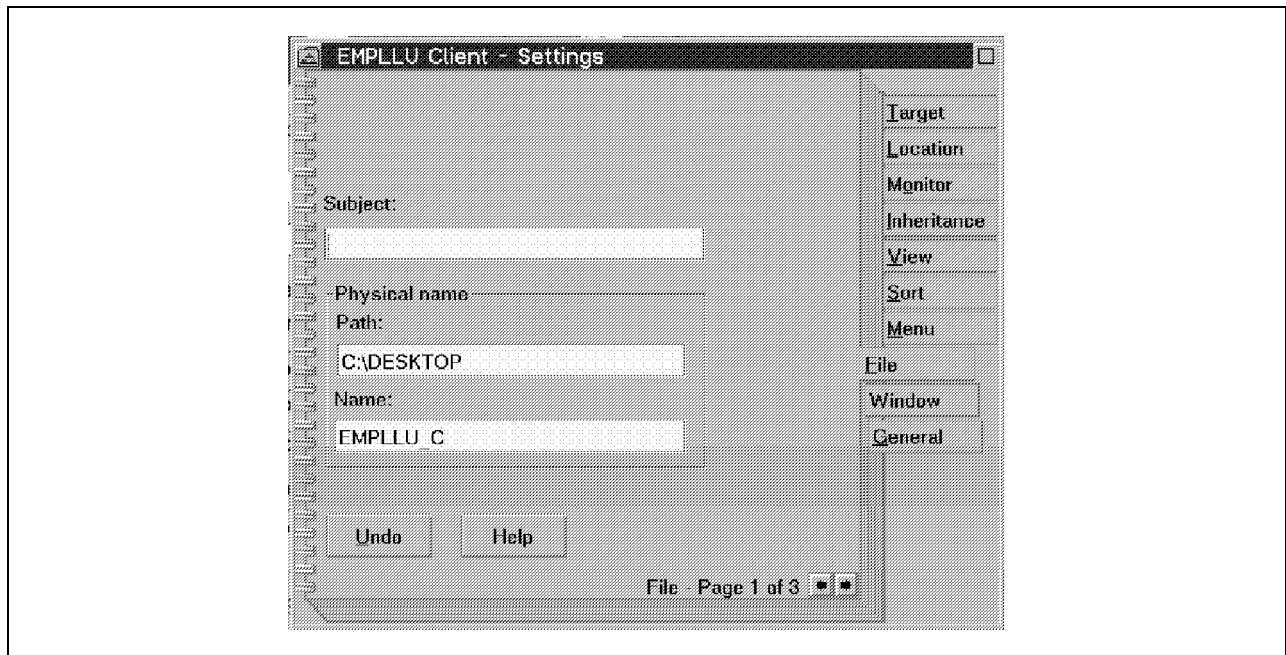


Figure 163. EMPLLU Client-Settings Window for Client File Settings

9. To close the settings for the client project, double-click on the **system menu** icon in the upper left-hand corner of the window.
10. Open an OS/2 window and copy the files for the client from the diskette to the hard disk using the following commands:

```
COPY A:\EMPLLU\CLIENT D:\IBMCOBOL\EMPLLU
COPY A:\EMPLLU D:\IBMCOBOL\EMPLLU
```

where D is the drive on which you have installed IBM VisualAge for COBOL for OS/2. Press Enter after each command.

Open the EMPLLU Server project by double-clicking on the **EMPLLU Client** icon on the Desktop. The EMPLLU Client-Icon view window appears (Figure 164 on page 100).

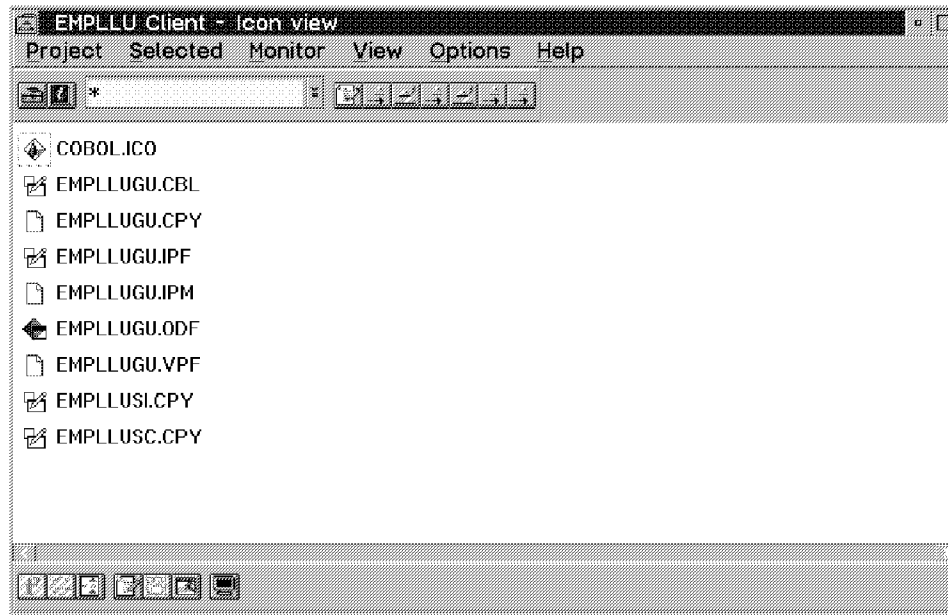


Figure 164. EMPLLU Client-Icon view Window

The settings for the client program are now complete.

11. To compile the client project, click on **Project** on the menu bar and select **Build** from the pull-down menu.

After the successful completion of the build command, prepare your environment to run the application.

Note

If you install the client and the server part of the application on the same computer, you should install the client part in the D:\IBMCOBOL\EMPLLU\CLIENT directory to ensure a clear separation of the client and the server.

The copy file EMPLUSI.CPY, which is used from both parts, need not be installed twice. Install it only in the D:\IBMCOBOL\EMPLLU directory, which can be shared by both parts. As a result, you have to specify the SYSLIB environment variable for the client part in the same way as in the server part to find the copy file in the D:\IBMCOBOL\EMPLLU path. If you need help to do this, look for the description of Figure 153 on page 92.

2.3.2.3 Running the Application

To run the program you have compiled, you first need to do the following:

1. Copy the D:\IBMCOBOL\EMPLLU\SERVER\EMPLLUSV.DLL file into the D:\IBMCOBOL\DLL directory.
2. Start CICS on the CICS server machine. Open an OS/2 window and type

CICSRUN

or double-click on the **CICS** icon on the Desktop and then double-click on the **Start CICS** icon.

3. To log on to the database server, open an OS/2 window on the DB2 server machine and type

DB2START

then press Enter. When the logon window appears, type the user ID and the password you defined as administrator in the appropriate fields. In our example, we used CSCBADM for the administrator ID and PASSWORD for the password.

4. On the client, open the EMPLLU Client project folder and select **Project** on the menu bar, then select **Run**. The empty Employee Lookup window appears (Figure 165). Click on the **Display** button. The client program calls the server program using an ECI call, the server program accesses the database, gets the data, calls the service routine to calculate, and hands over all data to the client program. It displays the data in the Employee Lookup window.

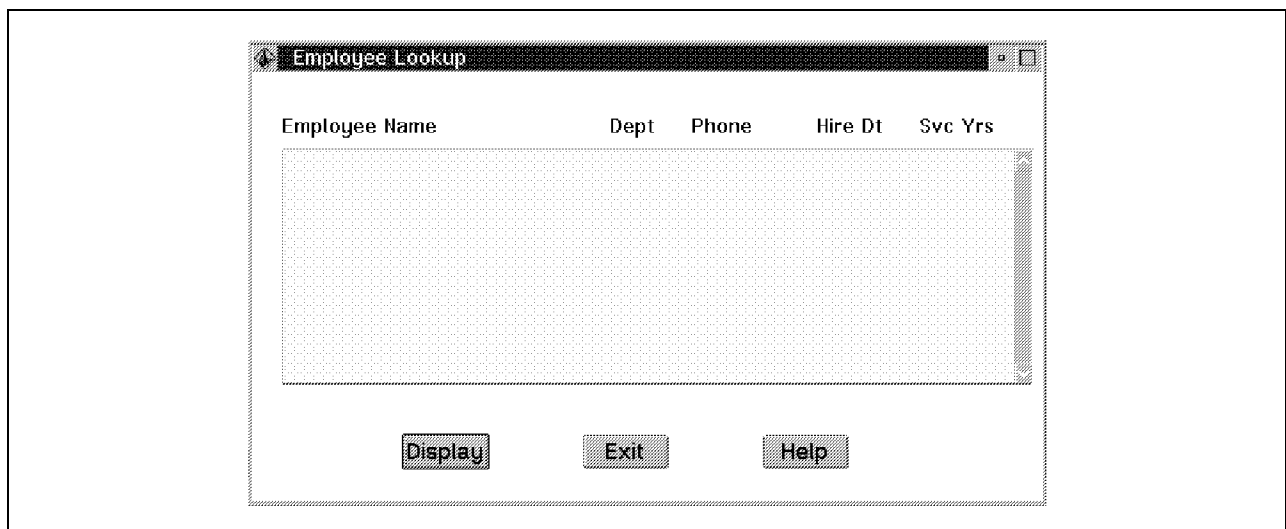


Figure 165. Employee Lookup Window

5. If the CICS server workstation is not yet logged on to the database server node, the Node Logon window appears when the server program issues the CONNECT TO SAMPLE command (Figure 166). On this window, the node CSCBS1 is already specified because it has been defined in the DB2 Client Setup as the node where the SAMPLE database is located. Type the DB2 administrator's user ID and password in the corresponding fields.

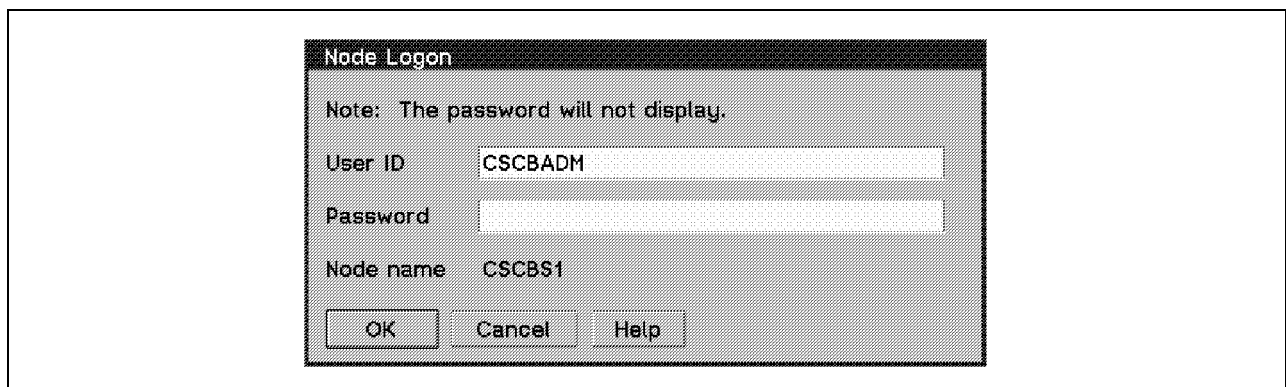


Figure 166. Node Logon Window

Alias

For the connection to the database server, use the user ID with which the SAMPLE database was installed: the system administrator ID. Only the system administrator has authority over all databases managed. If you cannot log on with this user ID, you must create a synonym or alias for the user ID you want to use. To create an alias that does not match the current authorization ID, system administrator authority is needed. The syntax for this alias or synonym statement is as follows:

```
CREATE ALIAS alias-name FOR table-name
```

In our case the alias-name is SAMPLE and the table-name is CSCBADM.SAMPLES where CSCBADM is the administrator ID.

Compile

The **build** command fails when you rebuild your server part of the application after the application has run. The build function stops before the DB2 precompile can run because CICS still holds the application. To prevent this, first stop the calling application on the client machine, then stop CICS on the server machine. When CICS sets the application free, the build continues.

When the compile error message `an error occurred while reading sidefile2` occurs, open the compiler options by selecting **Options** in the menu bar of the project and then **Compile** from the pull-down menu. Select the **Processing** tab to reach the page with the compiler processing control options. Deselect **Generate SYSADATA**. The current value changes to NOADATA and a message window appears. Click on **OK**.

The effect of NOADATA is that you can no longer jump into the source code by double clicking on an error message.

After each compile, copy the
D:\IBMCOBOL\EMPLLU\SERVER\EMPLLUSV.DLL file into the
D:\IBMCOBOL\DLL directory where D is your drive on which IBM VisualAge for COBOL for OS/2 is installed.

If you forget this copy, you receive a timestamp error. The error message in the client program reports a DB2 error as well.

2.3.3 How to Use the Transaction Assistant

The program named EMPLLU is divided into a client and a server part. The client part calls the server part via an CICS ECI call. To write this ECI call, you get support from the Transaction Assistant included in IBM VisualAge for COBOL for OS/2.

The Transaction Assistant simplifies the task of constructing a call to a CICS program for your non-CICS application. Transaction Assistant takes information you enter and generates the appropriate CALL statement. It then inserts the

CALL into your COBOL code. Optionally, you can have Transaction Assistant generate a COPY file for the CALL statement parameters.

At run time, the CALL statement in your client program calls the Transaction Assistant routine ECICALL, which calls the CICS for OS/2 ECI facility. The CICS ECI facility then invokes a server program that can be running either on the same computer or another computer.

To get more information about an ECI Call, see the box at the beginning of Chapter 2.2, "OS/2 CICS Client" on page 69.

To use the Transaction Assistant follow these steps:

1. Open the COBOL program code in which you want to call the CICS program, take place the cursor, and activate the Transaction Assistant by either clicking on the appropriate icon in the toolbar or select **Insert Code** from the Edit menu and select **CICS ECI** from the menu choice. The Transaction Assistant window appears (Figure 167).

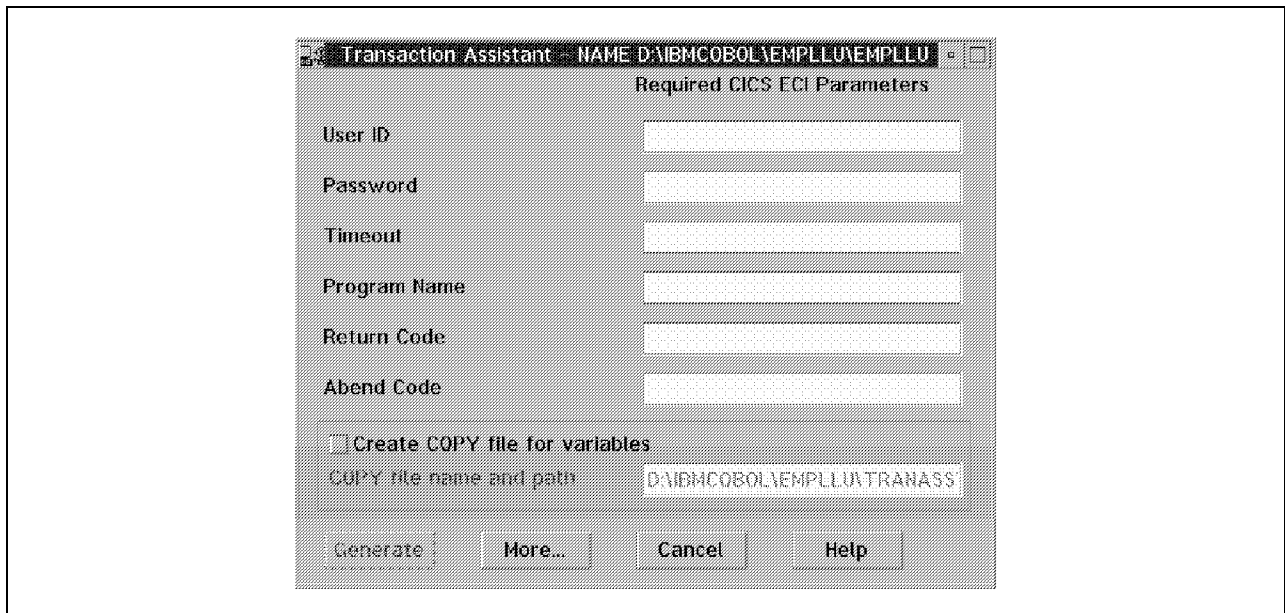


Figure 167. Transaction Assistant - Required CICS ECI Parameters Window

2. The parameters in this window are all required and to add values is necessary. The best way is to fill variables into these fields and to specify them in the program. This simplifies further changes of the source code. Type in the following variables:

ECI-USERID	in the User ID field
ECI-PASSWORD	in the Password field
ECI-TIMEOUT	in the Timeout field
ECI-PROGNAME	in the Program Name field
ECI-RETURN-CODE	in the Return Code field
ECI-ABEND-CODE	in the Abend Code field

as shown below (Figure 168 on page 104). Then click on the **More...** button.

Figure 168. Transaction Assistant - Required CICS ECI Parameters Window Filled

3. The window with the optional fields appears and it is advisable to define variables for the communication area and its length, too:

ECI-COMMAREA in the **CommArea** field and
ECI-COMMAREA-LENGTH in the **CommArea Length** field.

Then click on **OK** to save the parameters (Figure 169).

Figure 169. Transaction Assistant - Optional Fields Window—CICS ECI Parameters

4. On the window containing the required parameters, click on **Generate** to insert the generated code by the Transaction Assistant into the source code of your program. The following ECICALL will be inserted in the code:

```
* Beginning of code generated by Transaction Assistant
CALL "ECICALL" USING
    BY REFERENCE ECI-RETURN-CODE,
    BY REFERENCE ECI-ABEND-CODE,
    BY REFERENCE ECI-USERID,
    BY REFERENCE ECI-PASSWORD,
    BY REFERENCE ECI-COMMAREA,
    BY VALUE ECI-COMMAREA-LENGTH,
```

```

        BY VALUE ECI-TIMEOUT,
        BY VALUE 0,
        BY REFERENCE ECI-PROGNAME,
        BY VALUE -1.
* End of code generated by Transaction Assistant

```

5. The only thing you have to do now for invoking the server program is to define the variables in the WORKING STORAGE SECTION of your program and to add the required values in the variables.

To do this, add the following code into the WORKING STORAGE SECTION:

```

01 ECI-WORK-FIELDS.
   05 ECI-RETURN-CODE      PIC S9(9) COMP-5.
   05 ECI-ABEND-CODE       PIC X(4) .
   05 ECI-USERID           PIC X(8) .
   05 ECI-PASSWORD         PIC X(8) .
   05 ECI-COMMAREA-LENGTH  PIC S9(9) COMP-5.
   05 ECI-TIMEOUT          PIC S9(9) COMP-5.
   05 ECI-PROGNAME         PIC X(8) .

```

After that, you either generate a separate file to define the communication area (commarea) or you include it directly in this code. Whether a separate commarea is useful or not depends on its size, but using an extra file for this definition is better programming style. Using a separate file, you have to add a COPY statement such as the following in the WORKING STORAGE SECTION, where EMPLPLUSI is the name of our commarea:

```

01 ECI-COMMAREA.
   COPY EMPLPLUSI.

```

ECI WORK FIELDS

The ECI-RETURN-CODE is the code returned by the CALL.

The ECI-ABEND-CODE is provided by the CICS program.

ECI-USERID should contain the user ID on the CICS system under which the CICS program will run, and the ECI-PASSWORD should be associated with the user ID.

In the ECI-TIMEOUT, you can define the value for the time-out of the communication in seconds.

ECI-PROGNAME should contain the name of the CICS program that you want to invoke.

ECI-COMMAREA is the CICS communications area used between the OS/2 client and the server and ECI-COMMAREA-LENGTH is its length.

The optional parameter TRANSACTION that we left out stands for the name of the CICS transaction under which the CICS program will run. The default transaction is used when the variable is left blank.

The default server is also used when the variable for the optional parameter CICS SERVER NAME is left blank. The variable stands for the name of the server where the CALL will be directed.

6. Finally, define the variables as we do here:

```

** Set fields needed to call CICS ECI **
MOVE 0 TO ECI-RETURN-CODE.
MOVE " " TO ECI-ABEND-CODE.
MOVE "SYSAD " TO ECI-USERID.
MOVE "SYSAD " TO ECI-PASSWORD.
MOVE 608 TO ECI-COMMAREA-LENGTH.
MOVE 60 TO ECI-TIMEOUT.
MOVE "EMPLLUSV " TO ECI-PROGNAME.

```

7. The communication on the client side is already finished. The server program needs the definition of the communication area for program invocation in the LINKAGE SECTION as follows:

```

01 DFHCOMMAREA.
   COPY EMPLLUSI.

```

We have now completed the settings to call the server program via CICS ECI call.

2.3.4 Debugging IBM VisualAge for COBOL for OS/2 CICS Programs

You can use the IDBUG debugging utility to debug the programs in the CICS environment. To debug your client programs and server programs, you must add the debugging options to your compile settings. Follow these steps to add the options:

1. Open the COBOL project windows of your client programs and server programs.
2. Select **Compile** from the Options pull-down menu to open the Compiler Options window.
3. Select the **Debug** tab to get to the Debug page.
4. Check the **Compiler generates debugging information** check box. The entry field against the check box becomes *TEST*.
5. Check the **Linker includes debugging information** check box. The entry field against the check box becomes */DEBUG*. After you set the compile options and build the program, the compiled program is ready for debugging.
6. To debug the server programs on the CICS server machine, ensure that there is only one nonfacility task. To do this, you will need to use CEDA to set the CICS resources. Set the following values in the corresponding fields of the System Initialization Table (SIT):

```

Task Control
Maximum number of tasks: 3
Minimum free tasks: 1

```

7. After setting the task control, start CICS for OS/2 by typing

```
CICSRUN /D-@02@(IDBUG)
```

from the OS/2 command line or creating a program icon to specify the following:

Path and file name:	C:\OS2\CMD.EXE
Parameters:	/K CICSRUN.CMD /D-@02@(IDBUG)
Working directory	d:\CICS300\RUNTIME

Click on the new icon. (The nonfacility task would have identifiers such as @02@.)

When CICS is started, the debugger will be invoked and the IDBUG—Debug session Control window will appear on the screen of the CICS server machine.

8. Start the client program by selecting **DEBUG** from the Project pull-down menu of your client program. The IDBUG window will appear, from which you can debug your client program. When the client program invokes the server program, another debugger window appears on the server machine and you can debug the server program there as well.

Note

The debugger stores all of the options you specify during a debugging session (such as break points, program variables to monitor, and so on) in specific initialization files. These files are used to set up your debugger session when you restart the debugger. Occasionally, the debugger shuts down as soon as you run the first step of your application program. In this case, you must delete the initialization files first and then start CICS again with the debugging option.

To delete the debugger initialization files, type the following command in an OS/2 window:

```
del c:\os2\*@*
```

where c: is the drive on which OS/2 is installed.

Chapter 3. COBOL with DB2 for AIX and DB2 Client

This chapter describes the steps to implement a COBOL client/server solution in an environment that consists of an RS/6000 system and a PC with the following products installed:

- One RISC/6000 machine
 - AIX V4.1
 - DB2 for AIX V2.1.1
- One PC
 - OS/2 Warp Connect V3
 - VisualAge for COBOL V1
 - IBM TCP/IP for OS/2
 - DB2 for OS/2 Software Developer's Kits V2.1

The overview of this environment is illustrated in Figure 170 on page 110.

The two machines are connected to each other through a token-ring LAN with TCP/IP protocol. But you can use other physical links other than token-ring, for instance, Ethernet or X.25.

There is a domain name server used in this sample environment. If you have no name server in your environment, remember to add the IP addresses and the corresponding host names of the machines to the `/etc/hosts` file.

You will execute the COBOL SQL program, SALESDEP, that you have used in Chapter 1, "OS/2 Local Area Network with DB2" on page 1 on the OS/2 machine; while in this chapter the SALESDEP program will access data from the database on the AIX machine. You can use the OS/2 machine that you have configured. Therefore, you only have to install and configure the required software products on the AIX machine.

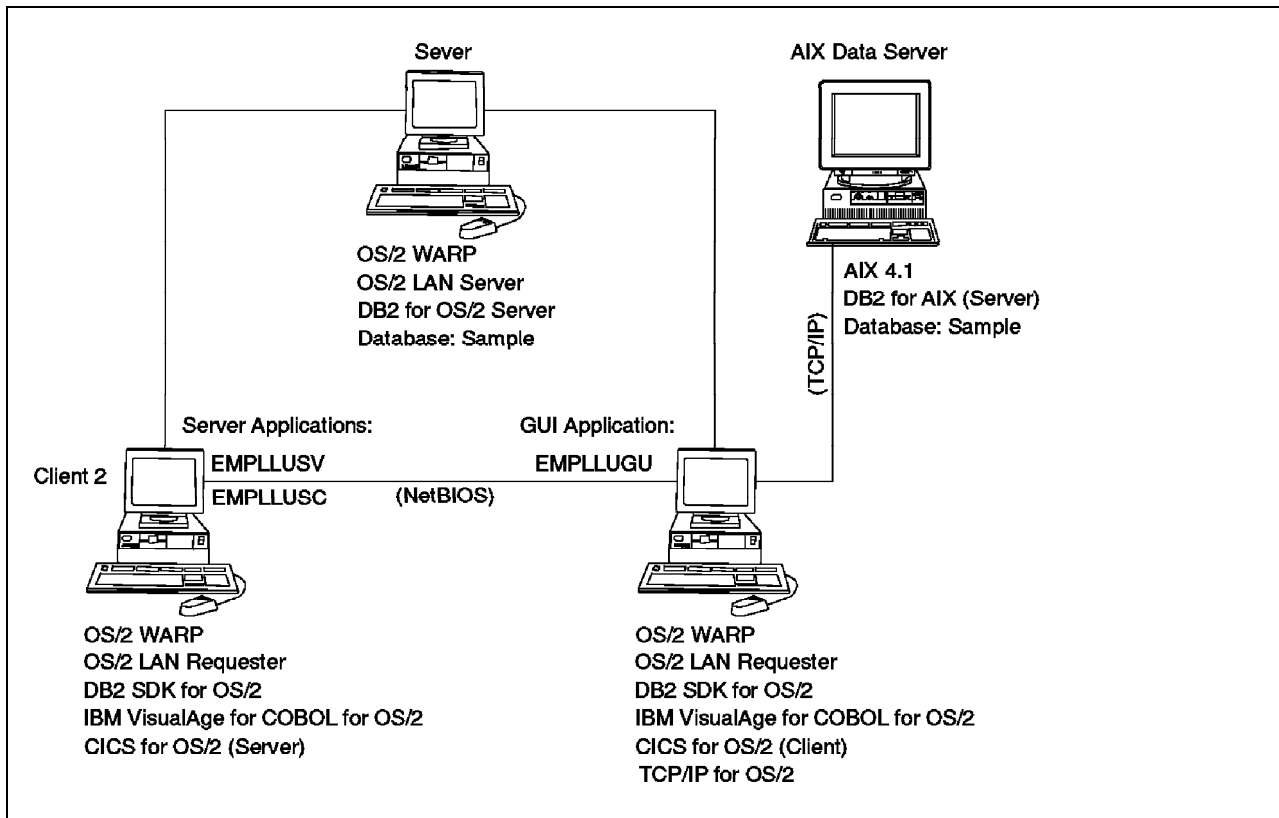


Figure 170. Overview of DB2 for AIX client/server environment

3.1 DB2 for AIX

DB2 for AIX is a member of DB2 family. DB2 for AIX products are categorized into several components as below.

- DB2 for AIX Single-User
- DB2 for AIX Server
- DB2 Client Application Enablers
- DB2 Software Developer's Kits and Administrator's Toolkits
- DDCS

DB2 for AIX Server includes the BASE DB2 engine functions and the code that enables DB2 to be accessed by local and remote clients. Remote clients must have the DB2 Client Application Enablers (DB2 CAE) installed to access a database server. If you need develop applications on the DB2 for AIX Server machine, you have to install DB2 SDK component on the server machine.

In this chapter, you need DB2 for AIX Server only. As you will compile COBOL SQL programs in the next chapter, you have to install DB2 SDK on the AIX as well.

3.1.1 DB2 for AIX Server Installation

To install DB2 products, follow these steps:

1. DB2 for AIX requires AIX 3.2.5 or higher level. However, as you will use IBM COBOL Set for AIX in Chapter 4. and it requires AIX V4.1.3 or above. It is recommended that you install AIX V4.1.3 on your AIX machine. To confirm that your AIX system level is V4.1.3 or higher, type

```
/usr/bin/oslevel
```

If the returned system level is lower than V4.1.3, upgrade your operating system and then continue the following steps.

2. Install the product files to the AIX machine, then

Log on as root.

Insert the DB2 for AIX product into the right-hand drive.

Enter the SMIT command and select the items listed below:

```
smitty install_latest
```

Install Software Products at Latest Level panel appears. Press F4 to get the INPUT device /directory for software panel. Move cursor to the device you use and press Enter key to select it (Figure 171).

```

                                Install Software Products at Latest Level

Type or select a value for the entry field.
Press Enter AFTER making all desired changes.

                                [Entry Fields]
* INPUT device / directory for software          []

+-----+-----+-----+-----+
|                                     |
| INPUT device / directory for software |
|                                     |
| Move cursor to desired item and press Enter. |
|                                     |
| /dev/rmt0.1 (5.0 GB 8mm Tape Drive) |
| /dev/fd0 (Diskette Drive)           |
| /dev/cd0 (SCSI Multimedia CD-ROM Drive) |
|                                     |
| F1=Help          F2=Refresh          F3=Cancel |
| F10=Image        F10=Exit            Enter=Do  |
| F5| /=Find       n=Find Next         |
| F9+-----+-----+-----+-----+

```

Figure 171. Input Device/Directory for Software Panel

The panel labeled Install Software Products at Latest Level appears again with more entry fields. Press F4 to select the DB2 components you want to install. In the panel labeled Software to Install (Figure 172 on page 112), you can move the arrow keys to the desired item and press F7 to select. You

have to install the following items to run your COBOL client/server programs discussed in this chapter:

- DB2 COBOL Language Includes Files and Samples
- DB2 Client Application Enabler
- DB2 Command Line Processor
- DB2 Communication Support - Base with TCP/IP
- DB2 Executables
- DB2 SDK Utilities and Samples
- DB2 Utilities and Samples

Or you can select DB2 Products Bundle to install all of the DB2 components.

```

                                Install Software Products at Latest Level
-----+-----+-----+-----+-----+-----+-----+-----+-----+
Ty+-----+-----+-----+-----+-----+-----+-----+-----+
Pr|                                     SOFTWARE to install                                     |
|                                                                              |
| Move cursor to desired item and press F7. Use arrow keys to scroll. |
| *   ONE OR MORE items can be selected.                                |
| *   Press Enter AFTER making all selections.                          |
|                                                                              |
| [MORE...7]                                                              |
| #-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 2.1.1.1  db2                                                            |
|   + 2.1.1.1  DB2 Products Bundle                                       |
|                                                                              |
| 2.1.1.1  db2_02_01                                                      |
|   + 2.1.1.1  DB2 C Language Include Files and Samples                 |
|   + 2.1.1.1  DB2 COBOL Language Include Files and Samples             |
|   + 2.1.1.1  DB2 Call Level Interface Samples                         |
| [MORE...20]                                                            |
|                                                                              |
| F1=Help          F2=Refresh          F3=Cancel                         |
F1| F7=Select      F8=Image             F10=Exit                         |
F5| Enter=Do       /=Find                n=Find Next                     |
F9+-----+-----+-----+-----+-----+-----+-----+-----+

```

Figure 172. Software to Install Panel

Press Enter to install these DB2 components. It will take several minutes to install the components. When the installation is finished, you will see the status sign as "OK." If the status sign shows "failed," check the message that explains what causes the failure. Fix the problem and then install the components again.

3. Create a group for the instance owner. Type

smitty mkgroup

In the Group name field, type the group name of the instance owner. for instance, cscbgrp. In the field ADMINISTRATIVE group?, set the value to true(Figure 173 on page 113). Press Enter.

Add a Group			
Type or select values in entry fields. Press Enter AFTER making all desired changes.			
		[Entry Fields]	
* Group NAME		[cscbgrp]	
ADMINISTRATIVE group?		true	
Group ID		[]	
USER list		[]	
ADMINISTRATOR list		[]	
F1=Help	F2=Refresh	F3=Cancel	F4=List
F5=Reset	F6=Command	F7=Edit	F8=Image
F9=Shell	F10=Exit	Enter=Do	

Figure 173. Add a Group Panel

4. Create a user for the instance owner. Type

smitty mkuser

The panel labeled Add a User appears (Figure 174 on page 114). Type the name of the instance owner, for instance, cscbadm, in the User NAME field. Type the group name you defined in the previous step, for instance, cscbgrp, in the Primary Group field. Press Enter. The user will be created and it will be the instance owner.

Add a User			
Type or select values in entry fields. Press Enter AFTER making all desired changes.			
[TOP]	[Entry Fields]		
* User NAME	[cscbadm]		
User ID	[]		
ADMINISTRATIVE USER?	false		
Primary GROUP	[cscbgrp]		
Group SET	[]		
ADMINISTRATIVE GROUPS	[]		
Another user can SU TO USER?	true		
SU GROUPS	[ALL]		
HOME directory	[]		
Initial PROGRAM	[]		
User INFORMATION	[]		
EXPIRATION date (MMDDhhmmyy)	[0]		
Is this user ACCOUNT LOCKED?	false		
[MORE...31]			
F1=Help	F2=Refresh	F3=Cancel	F4=List
F5=Reset	F6=Command	F7=Edit	F8=Image
F9=Shell	F10=Exit	Enter=Do	

Figure 174. Add a User Panel

5. Create an instance

Type the following in the command:

```
/usr/lpp/db2_02_01/instance/db2instance instid
```

instid is the user name of the instance owner you defined in the previous step, for instance, cscbadm.. The instance will be created.

6. Create a link from system library. Type

```
/usr/lpp/db2_02_01/cfg/db2ln
```

7. Prepare the .profile for the instance owner to set the appropriate environment variables.

Log on as the instance owner. Type the following commands.

```
cp $HOME/sqlllib/db2profile $HOME/.profile
. ./profile
```

All the environment variables required by DB2 for AIX should be set correctly after .profile file is executed.

8. Start the DB2 to ensure that the installation and configuration is correct.

Under AIX prompt, type db2start. After a while, you will see a message to inform you that the database has been started successfully.

9. Run the db2sampl executable file provided by DB2 for AIX to create the Sample database that will be used in the SALESDEP program.

Type

```
/usr/lpp/db2_02_01/samples/db2sampl
```

After several minutes, a message will inform you that the SAMPLE database has been successfully created.

10. To ensure the database is correct, you can type the following command to verify:

```
db2 list database directory
```

You will see the list of the databases. SAMPLE database should be in the list (Figure 175).

```
$db2 list database directory

System Database Directory

Number of entries in the directory = 1

Database 1 entry:

Database alias           = SAMPLE
Database name            = SAMPLE
Local database directory = /home/cscbadm
Database release level   = 6.00
Comment                  =
Directory entry type      = Indirect

$
```

Figure 175. Database List

11. You also can type the command below to see the tables in the sample database:

```
db2 connect to sample
```

You will see the information as below (Figure 176).

```
$db2 connect to sample

Database Connection Information

Database product          = DB2/6000 2.1.1
SQL authorization ID      = ROOT
Local database alias      = SAMPLE

$
```

Figure 176. Database Information

12. If you type the following command,

```
db2 list tables
```

you see the tables of the connected database (Figure 177 on page 116).

```
$db2 list tables
```

NAME	CREATOR	TYPE	CTIME
DEPARTMENT	CSCBADM	T	1996-03-28-09.57.34.000397
EMP_ACT	CSCBADM	T	1996-03-28-09.57.35.000338
EMP_PHOTO	CSCBADM	T	1996-03-28-09.57.36.000755
EMP_RESUME	CSCBADM	T	1996-03-28-09.57.38.000078
EMPLOYEE	CSCBADM	T	1996-03-28-09.57.34.000631
ORG	CSCBADM	T	1996-03-28-09.57.33.000541
PROJECT	CSCBADM	T	1996-03-28-09.57.36.000377
STAFF	CSCBADM	T	1996-03-28-09.57.33.000871

```
8 record(s) selected.
```

```
$
```

Figure 177. List of Tables with Information

You can also use some SQL statements to check if DB2 for AIX works correctly.

Now your DB2 for AIX should be ready to work.

3.1.2 DB2 for AIX Server Configuration

Run the SALESDEP program on your OS/2 client machine to access the SAMPLE database on the AIX system. The client/server configuration of DB2 is required. The communication protocols that support the connection between OS/2 and AIX are LU6.2 and TCP/IP. In this section, you use TCP/IP protocol to connect the DB2 for AIX server machine and the DB2 for OS/2 client.

3.1.2.1 TCP/IP Configuration

If you have never before configured the TCP/IP on the AIX machine, follow these steps. If you have already configured TCP/IP, you can skip these steps.

Type the SMIT command and select the items described below:

smitty tcpip

TCP/IP panel appears. Select **Minimum Configuration & Startup**, press Enter. Available Network Interfaces panel will be displayed.

Select the network interface adapter you use from the list (Figure 178 on page 117).

TCP/IP

Move cursor to desired item and press Enter.

Minimum Configuration & Startup
 Further Configuration
 Use DHCP for TCPIP Configuration & Startup

Available Network Interfaces

Move cursor to desired item and press Enter.

en0	Standard Ethernet Network Interface
et0	IEEE 802.3 Ethernet Network Interface
tr0	Token Ring Network Interface

F1=Help
F2=Refresh
F3=Cancel

F8=Image
F10=Exit
Enter=Do

F1| /=Find
n=Find Next

F9+

Figure 178. Available Network Interfaces

The panel labeled Minimum Configuration & Startup appears and you have to fill out all the fields (Figure 179 on page 118). You may have to ask your LAN administrator to assign you all the TCP/IP information. Set **Start Now** as **yes**, press Enter after you enter all the required fields. After the system completes the configurations and restarts the TCP/IP daemons, your AIX machine is ready to connect to another machine on this network.

Minimum Configuration & Startup			
To Delete existing configuration data, please use Further Configuration			
Type or select values in entry fields. Press Enter AFTER making all desired changes.			
		[Entry Fields]	
* HOSTNAME		[CSCBAIX]	
* Internet ADDRESS (dotted decimal)		[9.112.34.63]	
Network MASK (dotted decimal)		[255.255.252.0]	
* Network INTERFACE		tr0	
NAMESERVER			
Internet ADDRESS (dotted decimal)		[9.112.44.1]	
DOMAIN Name		[STL.IBM.COM]	
Default GATEWAY Address (dotted decimal or symbolic name)		[9.112.32.5]	
RING Speed		[16]	
START Now		yes	
F1=Help	F2=Refresh	F3=Cancel	F4=List
F5=Reset	F6=Command	F7=Edit	F8=Image
F9=Shell	F10=Exit	Enter=Do	

Figure 179. Minimum Configuration & Startup Panel

3.1.2.2 DB2 for AIX Configuration for TCP/IP

To use TCP/IP for the DB2 client/server configuration on the AIX system, follow these steps. We will describe the steps to configure the OS/2 DB2 client machine in the next section.

1. Log on as root.
2. Add two new services to the /etc/services file on your AIX system. The name of the services and the ports should be unique in the services file. The number of the two ports should be sequential. For instance,

```
db2tcPIP 4000/tcp
db2tcPIPI 4001/tcp
```

3. Set up the DB2 server as below.

Log on as the instance owner.

Update the service name in the database manager configuration.

```
db2 update database manager configuration using svcname servicename
```

servicename is the name of the first service port you defined in Step2, for instance, db2tcPIP.

4. Set the environment variable \$DB2COMM to TCPIP in the instance owner's .profile file. Type the following from the command line or add it to the /etc/environment file:

```
export DB2COMM=TCPIP
```


5. Start the database by typing **db2start**.

Your DB2 system should now be able to connect to its TCP/IP clients. After you finish the configuration of your OS/2 client machine, you can test the connection between the AIX system and the OS/2 system.

3.2 Set up DB2 Client to Access DB2 for AIX

You already have DB2 SDK for OS/2 components installed on your OS/2 machine, so you need only install TCP/IP for OS/2 on the OS/2 machine and set up the DB2 client configuration to access the database on the DB2-for-AIX server machine. This section describes how to install TCP/IP and configure the DB2 client.

3.2.1 TCP/IP Installation

In order to connect your DB2 client machine to your DB2-for-AIX server machine, use TCP/IP as the communication protocol. In 3.2, "Set up DB2 Client to Access DB2 for AIX," you configured the DB2 for AIX as the server in the TCP/IP environment. You now continue the setup for the DB2 client.

As you not yet installed any TCP/IP product on your DB2 client machine, you must install the IBM TCP/IP for OS/2 first.

Follow these steps to install IBM TCP/IP for OS/2:

1. IBM TCP/IP for OS/2 is included in OS/2 Warp Connect. Therefore, you can use your OS/2 Warp Connect product to install TCP/IP. Insert the OS/2 Warp Connect CD-ROM into the CD-ROM drive.
2. Double-click on the OS/2 System icon on the Desktop. The OS/2 System - Icon View window appears (Figure 180).

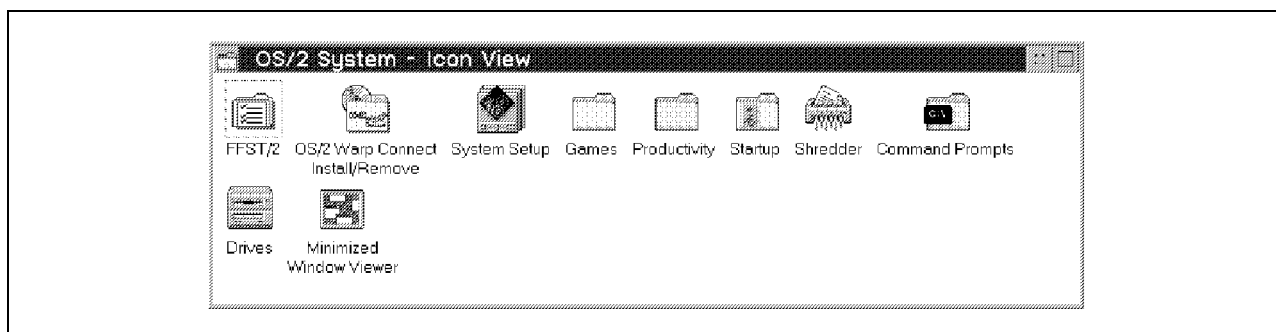


Figure 180. OS/2 System - Icon View Window

3. Double-click on OS/2 Warp Connect Install/Remove icon. OS/2 Warp Connect Install/Remove window will be displayed (Figure 181 on page 120).

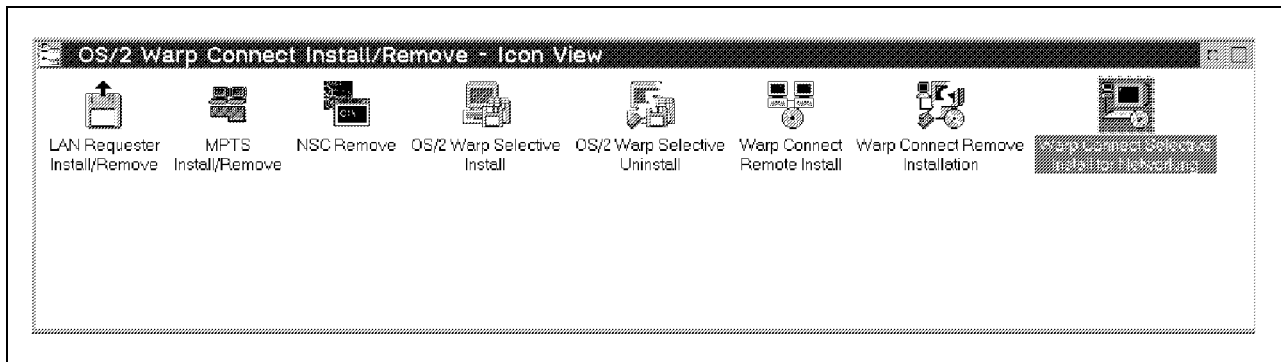


Figure 181. OS/2 Warp Connect Install/Remove - Icon View Window

4. Double-click on the icon labeled Warp Connect Selective Install for Networking. The OS/2 Warp Connect window appears. Click on **OK**, and the Local versus Remote window appears (Figure 182).

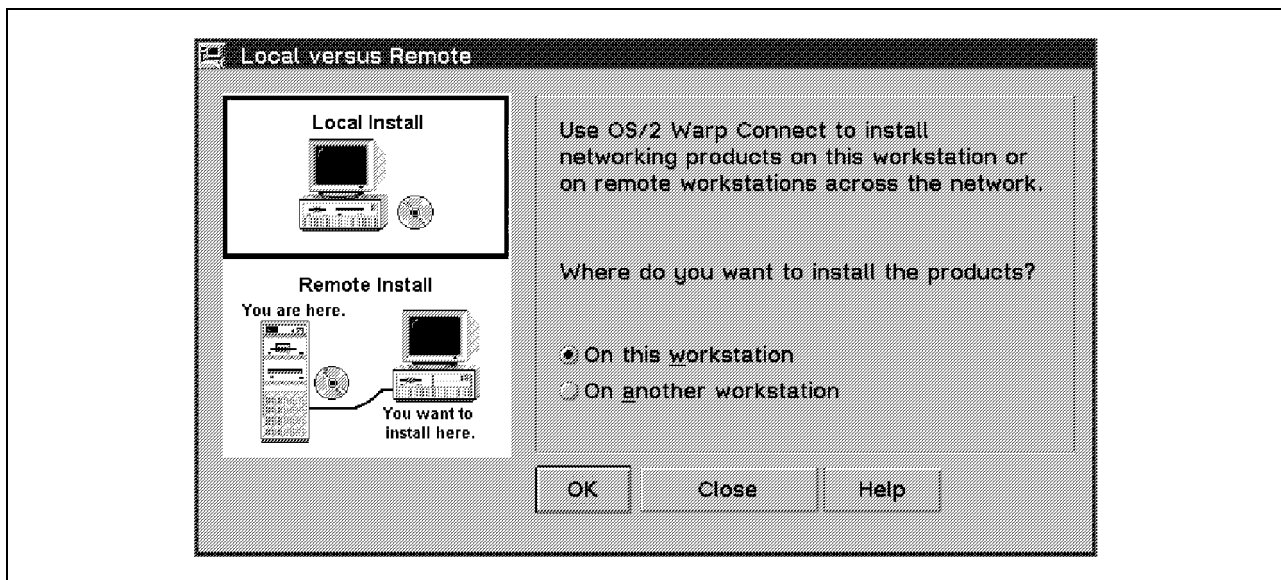


Figure 182. Local versus Remote Window

5. As you will install TCP/IP on the local machine, select **On this workstation**. Click **OK**. A window labeled Installing OS/2 Warp Connect appears as shown in Figure 183 on page 121.

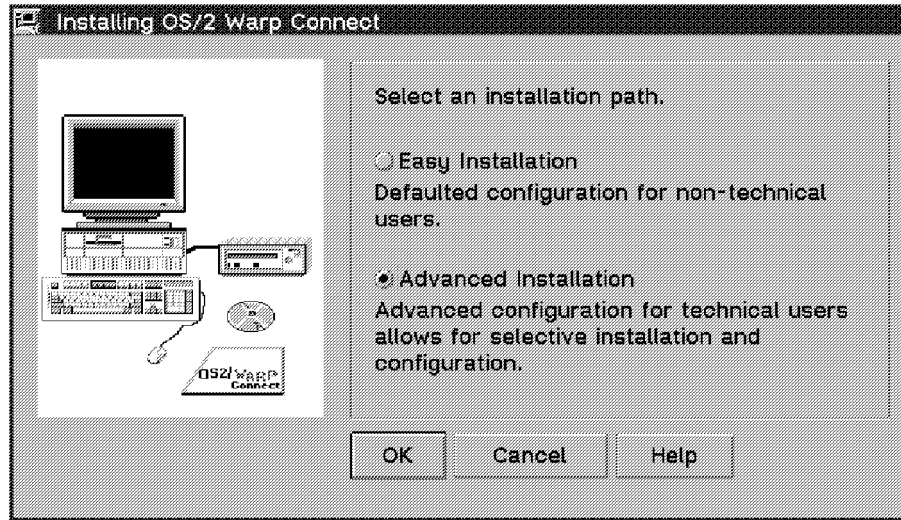


Figure 183. Installing OS/2 Warp Connect Window

6. There are two selections in the window. Because the Easy Installation does not install TCP/IP on the workstation, select **Advanced Installation**. Click on **OK**. The Product Selection window will be displayed (Figure 184).

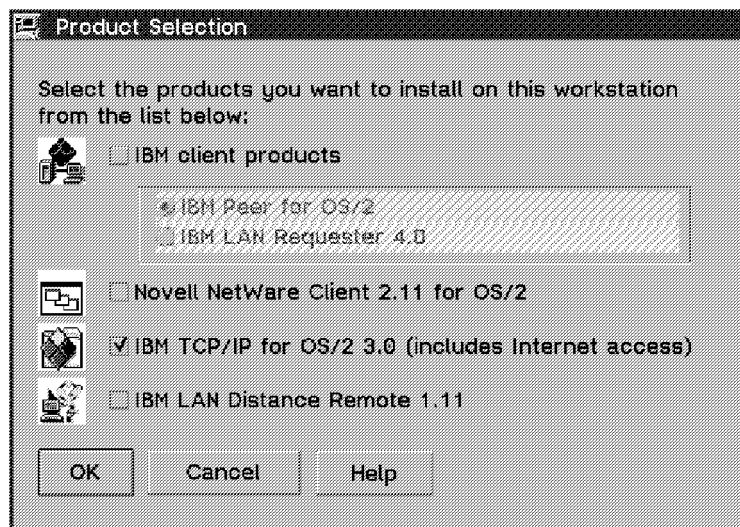


Figure 184. Product Selection Window

7. Check **IBM TCP/IP for OS/2 3.0 (includes Internet access)** check box. Click on **OK**. You will see the window labeled Set up selected products, as shown in Figure 185 on page 122. The window includes a notebook with two pages. In the Adapter page, it shows the Adapter card used by the machine. You can change the setting by clicking on the **Settings...** button.

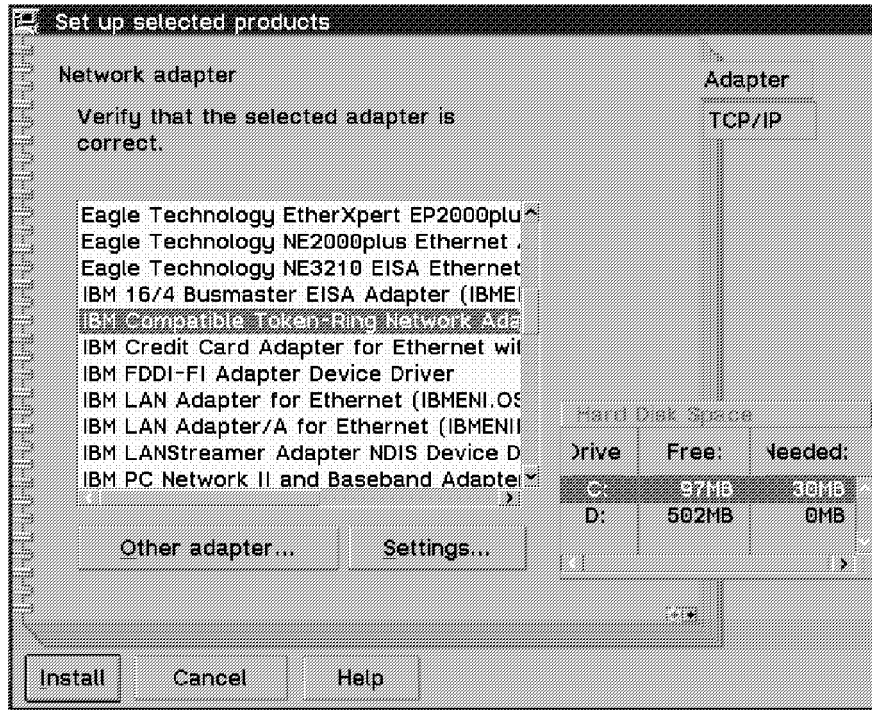


Figure 185. Set up Selected Products Window—Adapter Page

8. Click on **TCP/IP** tab to get to the TCP/IP page (Figure 186).

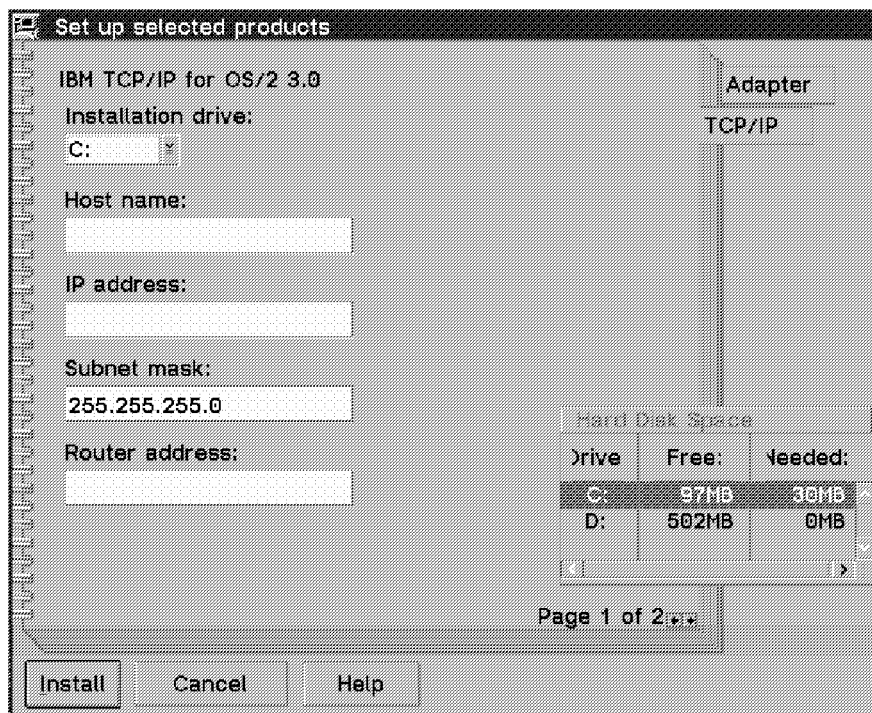


Figure 186. Set Up Selected Products—TCP/IP Page

In this page, you have to fill out the TCP/IP information of this machine. You have to ask the LAN administrator to assign the required TCP/IP information

for your machine. If you have your own LAN, you can set up these addresses by yourself, as long as they are unique in the network. Figure 187 on page 123 is an example of the settings in the TCP/IP page. After typing in all the required information in these fields, click on **Install**.

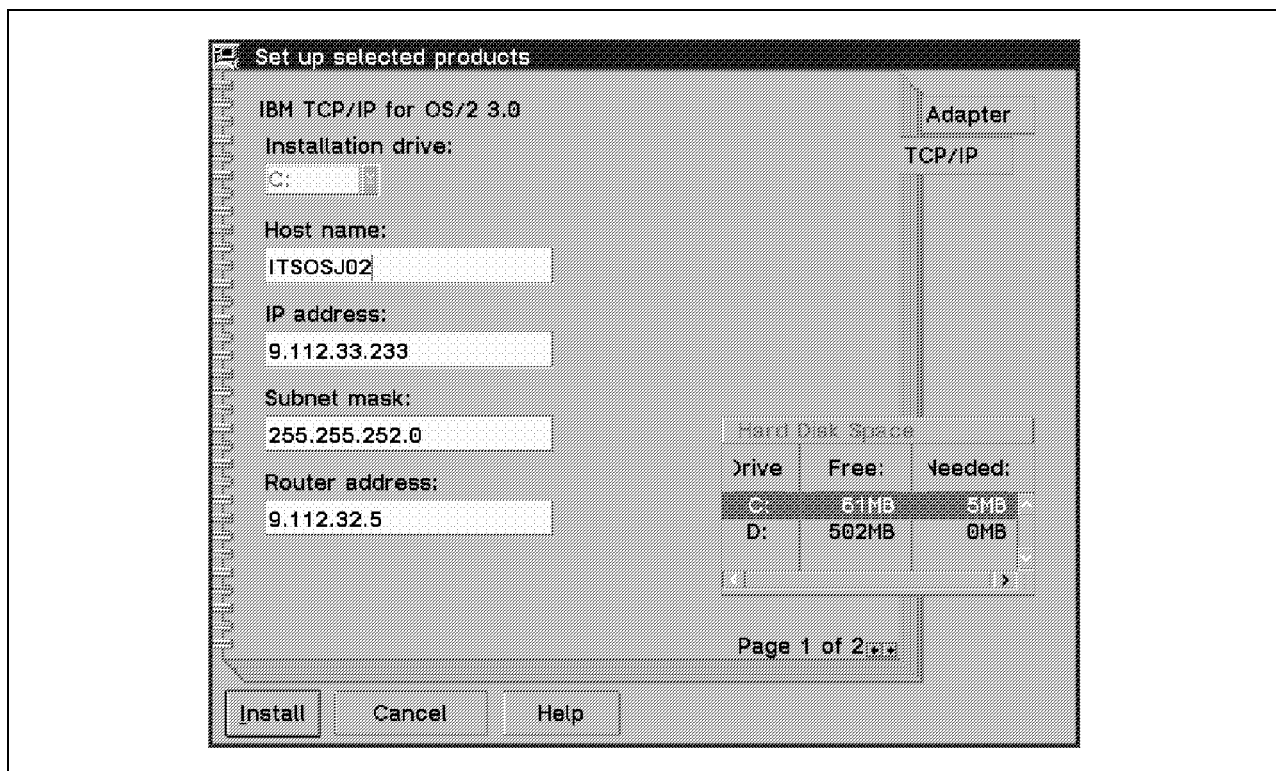


Figure 187. Set Up Selected Products—TCP/IP Page Example

9. A window labeled The setup is complete shows up to assure you that have set up all the information. If you want to change the settings, click on **Cancel** to return to the Product window. If you are sure that you have set up all the information correctly, press **Install** to start the installation.

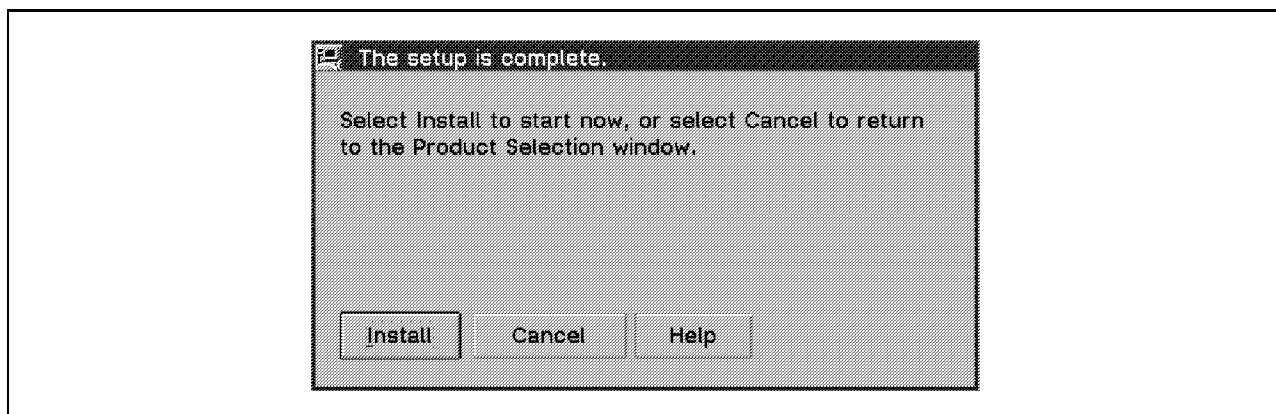


Figure 188. The Setup is Complete Window

10. The Installation program will copy a file to the system and then ask you to reboot the system. Reboot the system.

11. After you reboot the system, the Installation program will start the installation process. After a few minutes, you will see a message box to inform you the installation is complete.
12. You have to reboot the system to activate TCP/IP. But before you reboot your machine, you may have to change some settings for the TCP/IP configuration. Double-click on the **OS/2 System** icon and you will see the TCP/IP icon. Double-click on the **TCP/IP** icon. The TCP/IP - Icon View window appears (Figure 189)

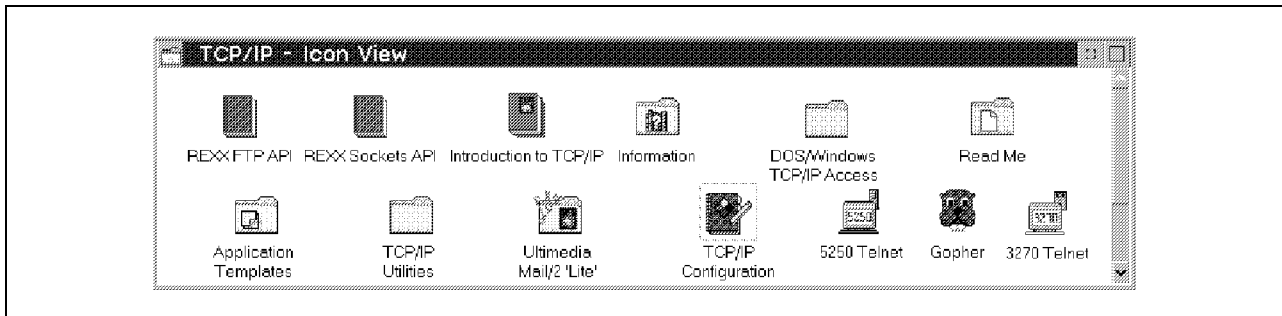


Figure 189. TCP/IP-Icon View Window

13. Double-click on the **TCP/IP Configuration** icon. You will see the TCP/IP Configuration window containing a note book. Click on **Hostname** tab and the Hostname page will be displayed (Figure 190).

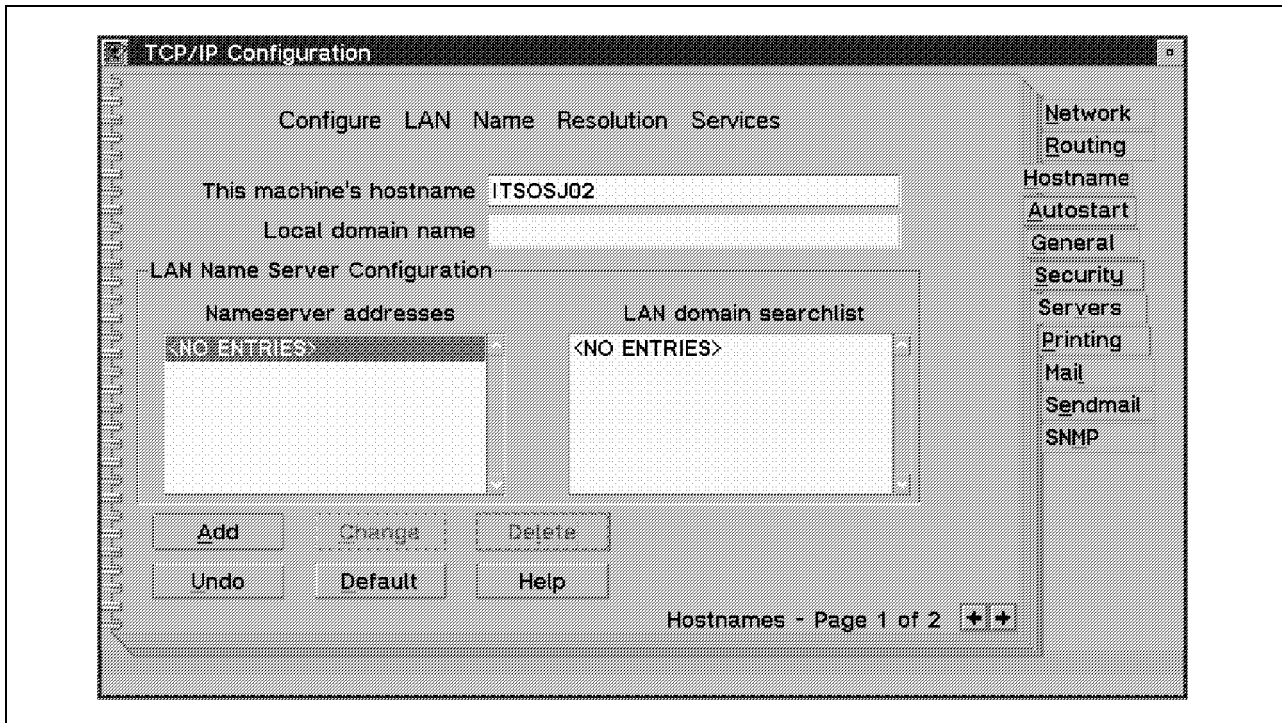


Figure 190. TCP/IP Configuration Window—Hostname Page

If you are using the Nameserver in the network, you have to add the address of the Nameserver. Click on **Add**. The Nameserver Entry window appears. Type the address of your Nameserver machine and click on **OK**. Close the TCP/IP Configuration window. The TCP/IP configuration of your machine should be ready.

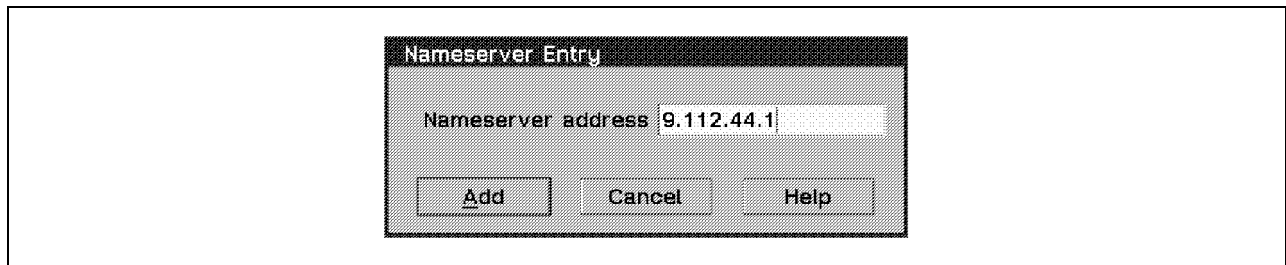


Figure 191. Nameserver Entry Window

14. Reboot your machine. You can test your machine to see if it works with TCP/IP protocol by typing the following command in the OS/2 command:

`ping hostname`

hostname is the host name of the DB2 for AIX server machine, in this case, CSCBAIX.

If you get the following response as shown in Figure 192, your machine is already connected to the DB2 server machine via TCP/IP. If you do not get any response, you have to check the configurations on both your client machine and server machine.

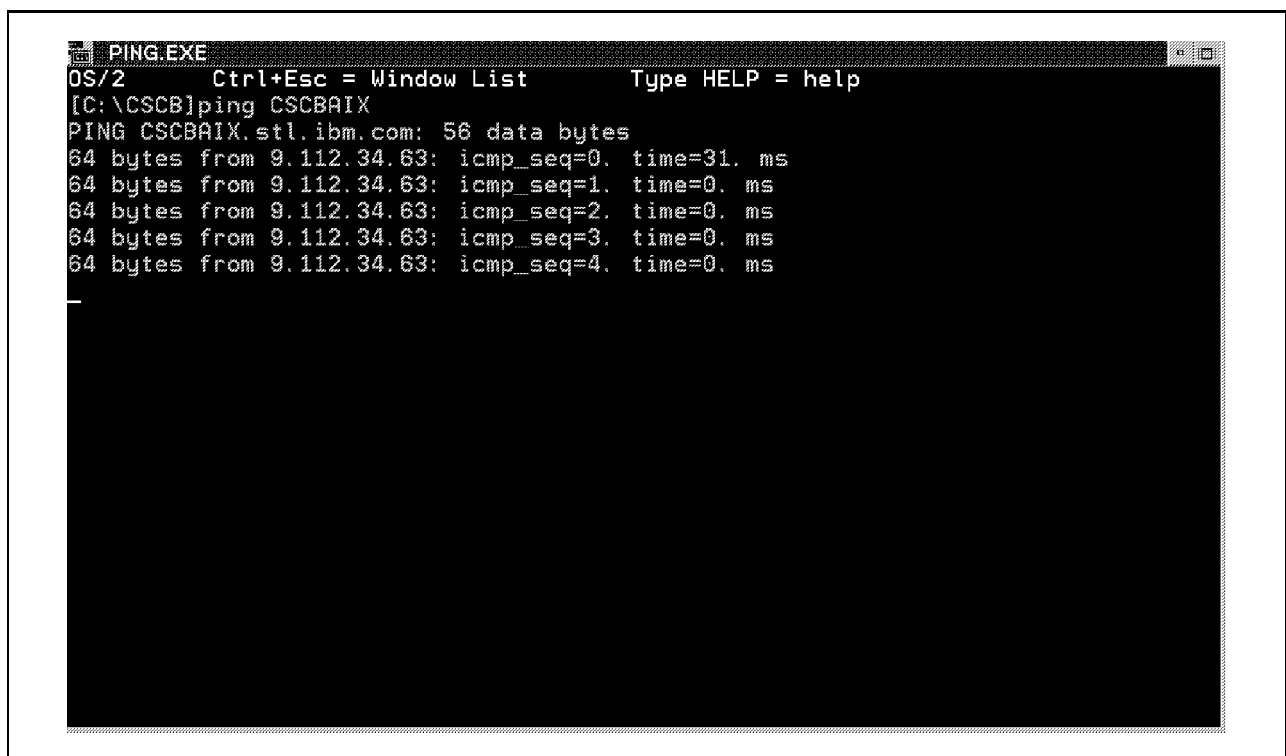


Figure 192. Response from DB2 for AIX server against Ping Command

3.2.2 DB2 Client Configuration

Once your client machine is connected to the AIX server machine, you can start to set up your client machine as the DB2 client of your DB2 for AIX server. This section describes how to configure your client machine to access the database on the DB2 for AIX server.

1. Double click on **IBM DATABASE 2** icon on the Desktop. IBM DATABASE 2 - Icon View window appears. Double click on Client Setup icon of this window. The DB2 Client Setup window appears (Figure 193 on page 126). Select **New** from the Node pull-down menu.

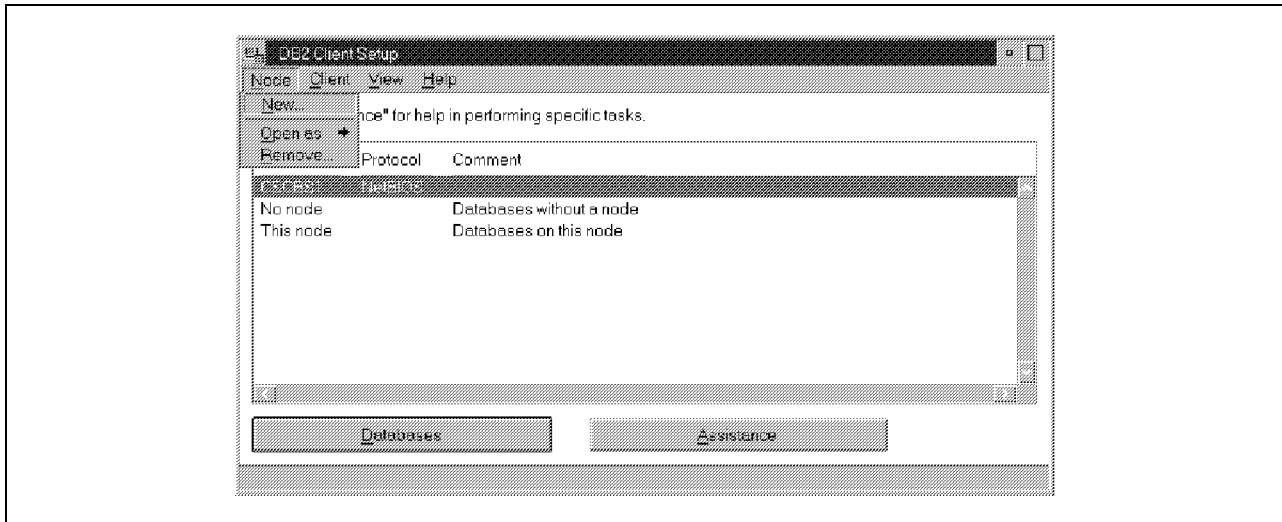


Figure 193. DB2 Client Setup Window

2. The New Node window appears Figure 194 on page 127 Type the alias you will use to specify the DB2 for AIX server machine in the Name field. Click on **TCP/IP** radio button to select the protocol. Type the hostname of the server machine defined in 3.2.1 "TCP/IP Installation" on page 119 in the Hostname field and also the name of the service port used by TCP/IP for the DB/2 connection. On the client machine, the service ports have to be the same as those defined in the AIX server. In this example, the service ports are as follows:

```
cscbtcpip      4000
cscbtcpipi     4001
```

Ensure that you have added the definition of the two ports in the `\mptn\etc\services` file of the client machine. Figure 194 on page 127 shows the sample entries of the New Node window.

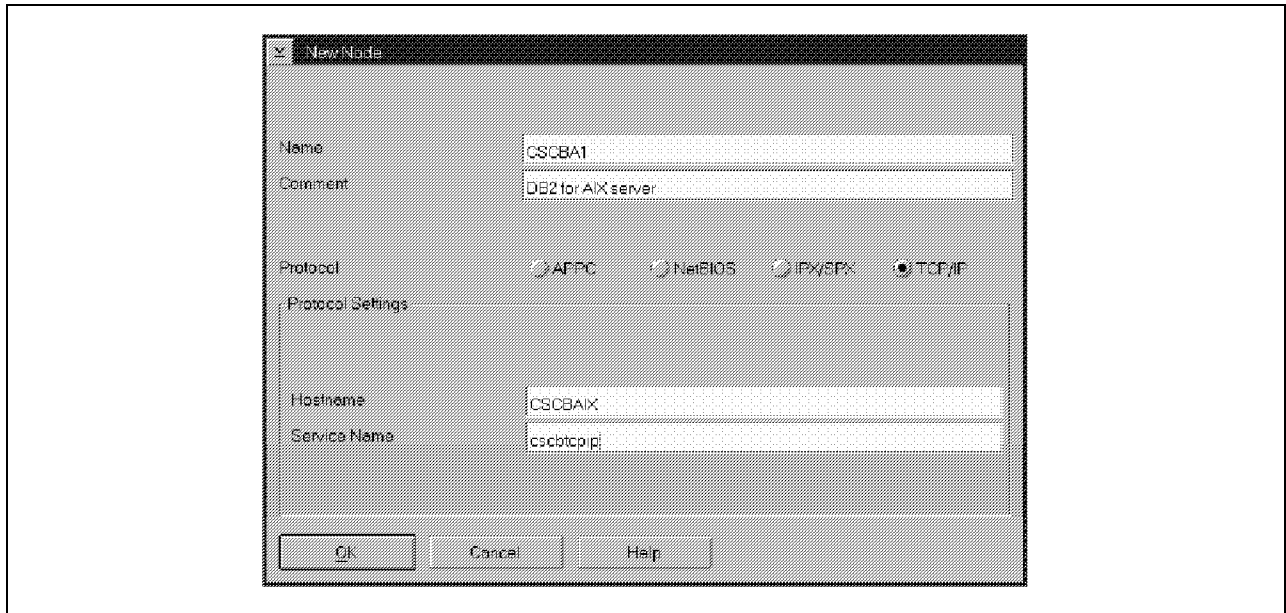


Figure 194. New Node Window

Click on **OK**. You will get back to the DB2 Client Setup window with the definition of the new node (Figure 195).

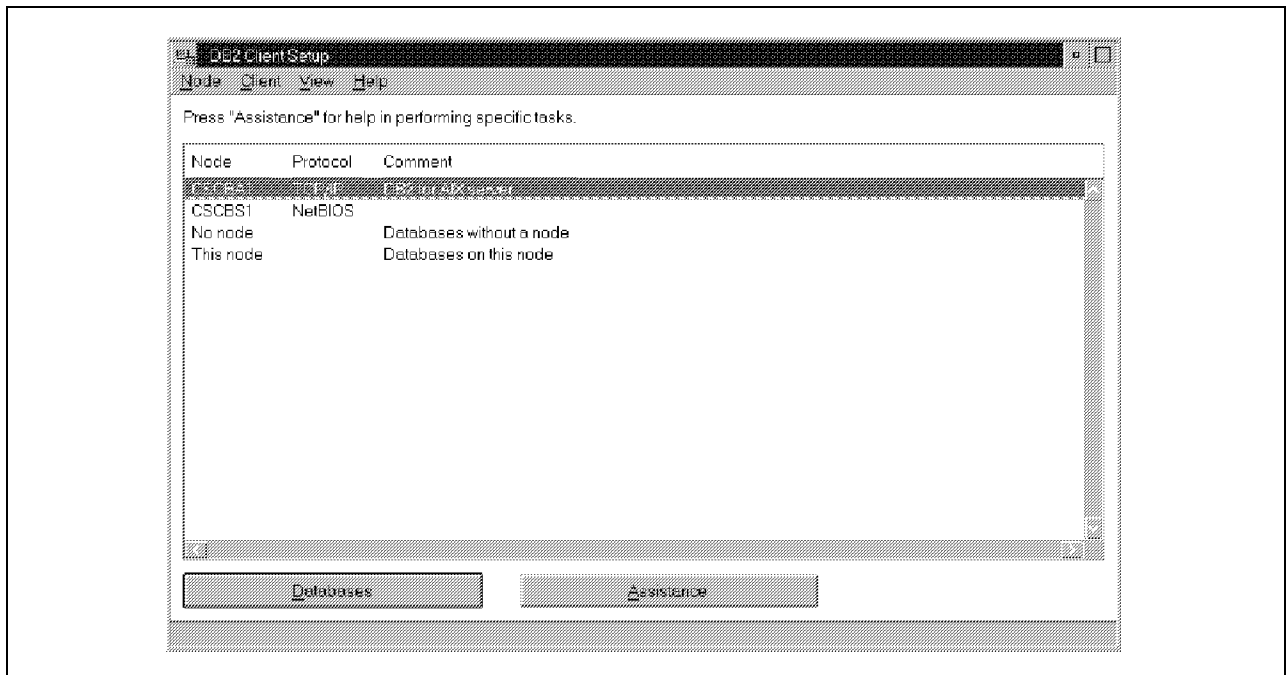


Figure 195. DB2 Client Setup Window with New Defined Node

- Double-click on the node item you just defined. For instance, double click on CSCBA1. You will see DB2 Client Setup - Database window (Figure 196 on page 128). In this window, you can catalog the remote database on the server machine.

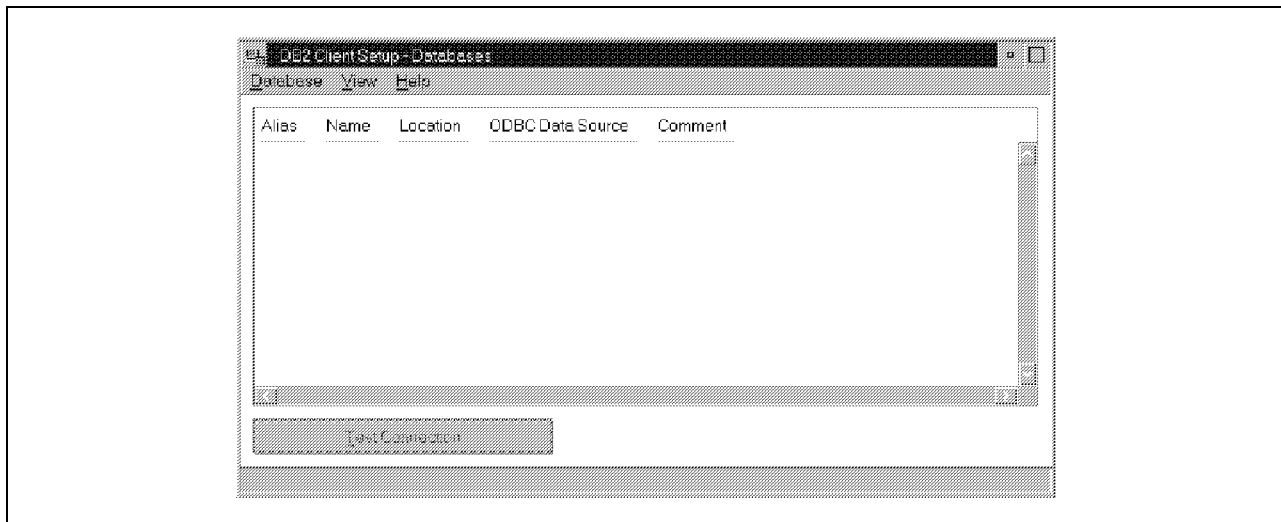


Figure 196. DB2 Client Setup-Database Window

4. Select **New** from the Database pull-down menu. You can then catalog the database that the DB2 client machine will access. In this section, you learn to access the SAMPLE database on the AIX server machine. (You already cataloged the SAMPLE database on the DB2 for OS/2 server machine.) Use an alias other than SAMPLE. Type the alias for the SAMPLE database on AIX server and the alias of the AIX server machine defined in the previous step (Figure 197).

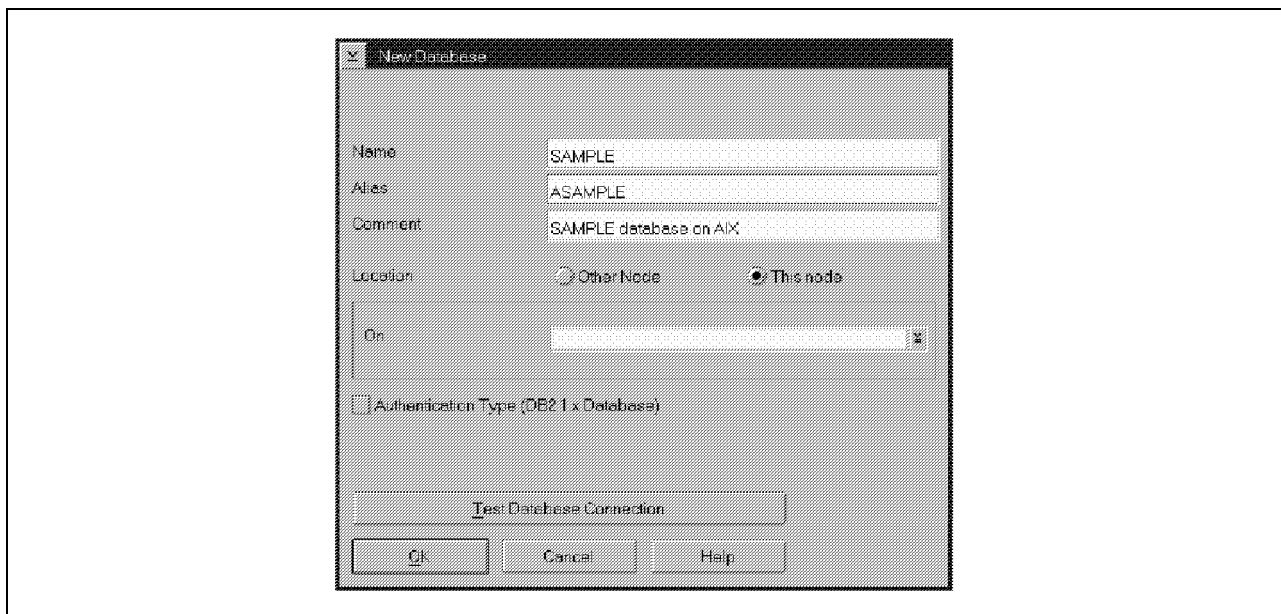


Figure 197. New Database Window

5. After typing all the required information, you can test the connection between the DB2 client and the DB2 server by clicking on **Test Database Connection**. The Client Application Enabler/2 window will be displayed to ask you to type the appropriate user ID and password. Since you are connecting to the database on the server, you have to log on with a user ID that has the authority to access the database on the AIX. Only an authorized user of the database can connect to the database. A message will inform you that the connection is successful.

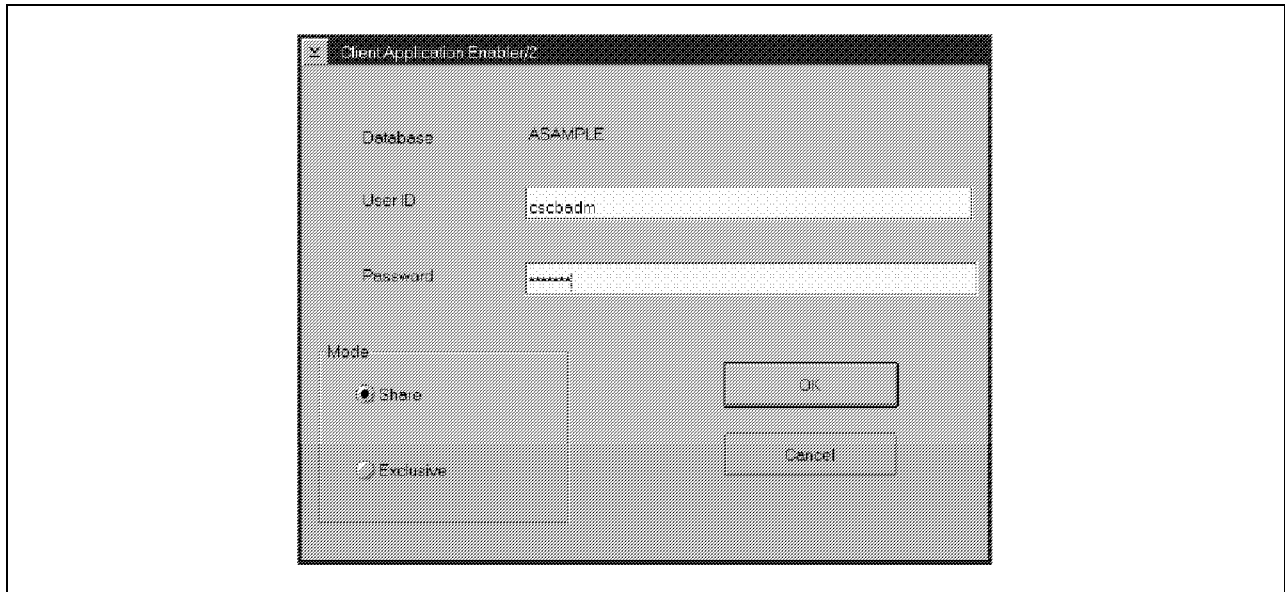


Figure 198. Client Application Enabler/2 Window

Note

If you log on the AIX server through the logon panel and the user ID you used to log on comes with a password in lower case characters, it is very likely that you will fail to log on. OS/2 Warp will convert all the input characters in the panel to upper case, but AIX treats lower case and upper case differently. Therefore, the password converted to upper case cannot be correct on the AIX server. To work around the problem, you should change the authorized user's password to upper case on AIX.

You can try to connect to the database, sample, that you just cataloged using the command under OS/2 command line:

```
db2 connect to sample
```

Once you log on to the server successfully, enter the following command:

```
db2 list tables
```

If you receive the same result as Figure 177 on page 116, you are ready to execute the sample program to access the sample database on the AIX server. If you do not get the correct result, you will have to check your setting again.

3.3 Executing a COBOL Program That Accesses Database on AIX

You will modify the same program, SALESDEP, that you have used in Chapter 1, "OS/2 Local Area Network with DB2" on page 1. For detailed information regarding the operations to build the program, please refer to Chapter 1, "OS/2 Local Area Network with DB2" on page 1. Because you configured the alias of the SAMPLE database on the DB2 for AIX server as SAMPLE, you must modify one SQL statement in the SALESDEP program. Open the SALESDEP project on the Desktop. Open the SALESDEP.CBL file, and move cursor to the SQL statement as below.

```
EXEC SQL
    CONNECT TO SAMPLE
END-EXEC.
```

The alias of the database you are connecting is `SAMPLE`, so you have to change the above statement into

```
EXEC SQL
    CONNECT TO SAMPLE
END-EXEC.
```

Change the coprocessor option by clicking the options menu and selecting the compile item. Click on the prep page of the note book. Check the box and modify the statements as below:

```
database sample bindfile package
```

After modifying this program, save the file. Before you start building this application, make sure whether the DB2 for AIX server is already started. If it is not yet started, issue `db2start` on the server machine to start the DB2. Select **Build** from Project pull-down menu to build the SALESDEP application. Once the application is successfully built, you can execute the SALESDEP program that accesses the `SAMPLE` database from the DB2-for-AIX server.

Chapter 4. COBOL with CICS for AIX and CICS Client for OS/2

This chapter describes the steps to implement a COBOL client/server solution in a environment that consists of the following systems and the corresponding software products:

- One RISC/6000 machine
 - AIX V4.1
 - DB2 for AIX V2.1.1
 - DB2 SDK for AIX V2.1.1
 - CICS for AIX V2.1
 - IBM COBOL Set for AIX V1.2
- One PC
 - OS/2 Warp Connect V3
 - VisualAge for COBOL V1.2
 - IBM TCP/IP for OS/2 V3
 - CICS Client for OS/2 V2

The levels of the software products used here are not necessarily the same as those used in your environment. Check the compatibility of the levels of the products before you start setting up your environment.

The overview of this environment is illustrated in Figure 199.

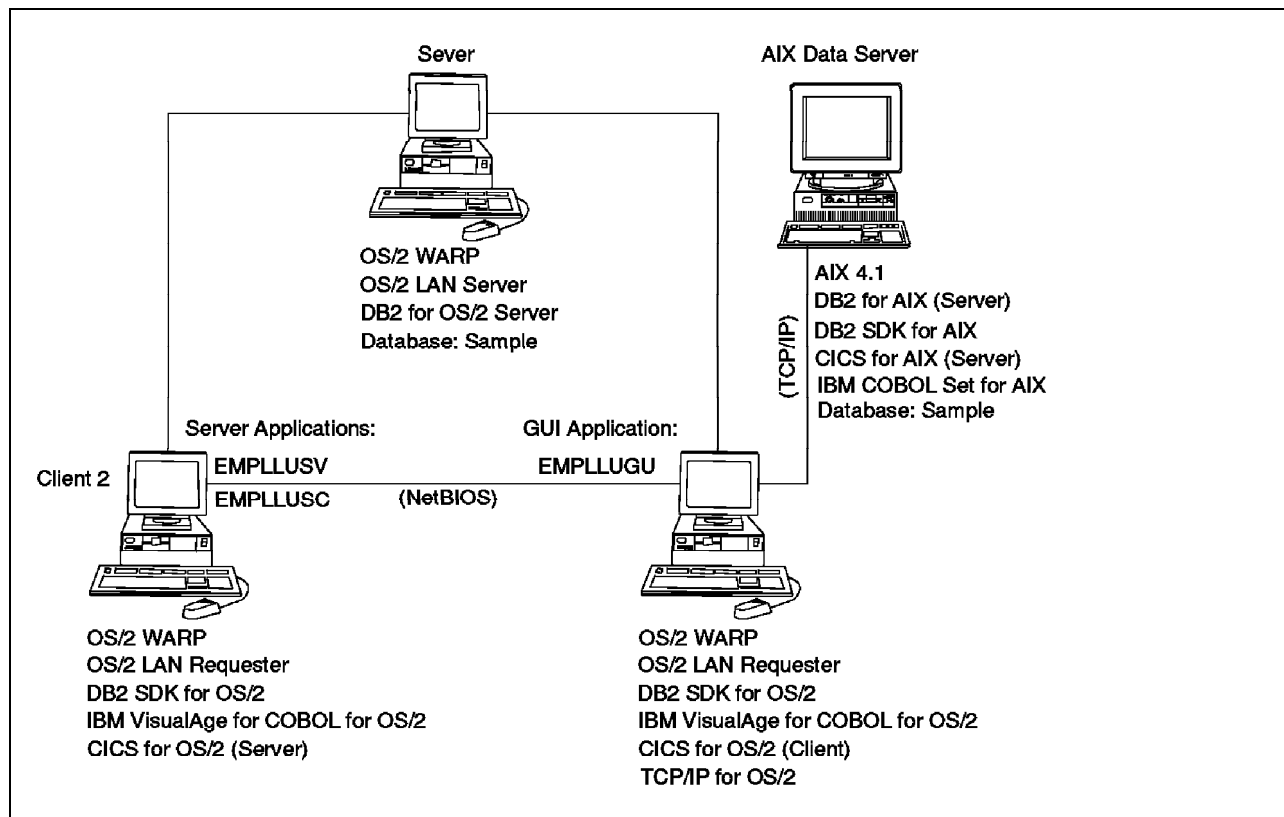


Figure 199. COBOL with CICS for AIX - CICS Client for OS/2

You have all the required software installed on your PC, as well as the AIX Operating System, DB2 for AIX, and DB2 SDK for AIX on your RS/6000; now you need only install CICS for AIX and COBOL Set for AIX on the RISC/6000 machine.

The aim is to execute the Employee Lookup application that you used in Chapter 2, “OS/2 LAN with CICS and DB2” on page 59, in this CICS Client/Server environment. The same client program, EMPLLUGU, runs on the same CICS Client machine while the server program executes on the CICS for AIX server machine. You do not even have to modify the server source programs and you can compile and link them on the AIX server. High portability and easy migration are the major benefits of the solutions based on IBM's CICS, COBOL and DB2 products that will be illustrated in this chapter.

The following sections describe the installation of the additional software products and the configuration for the CICS client/server environment.

4.1 CICS for AIX Server Setup

CICS for AIX is a member of the CICS family of products and it is also categorized in the group of CICS on Open Systems. CICS for AIX offers CICS capabilities for transaction processing on IBM RS/6000. It can communicate with other CICS systems as well as CICS Clients. Non-CICS client programs communicate with CICS for OS/2 server programs via ECI, EPI, or some other interfaces. It is also true when the server programs are in the CICS for AIX environment. Therefore, the same client programs can work with the corresponding server programs on different platforms. CICS family products provide a very good portability to the client/server solutions based on CICS across platforms from mainframes to workstations. Combined with the high compatibility of IBM COBOL family and DB2 family across multiple platforms, users can easily migrate their applications to the appropriate platforms and maintain a good scalability of their applications as well.

Follow the instructions described in this section and you will learn that you can easily migrate your server programs from OS/2 to AIX. These sections describe how to install IBM COBOL Set for AIX and how to set up your CICS for AIX server environment that the client programs on OS/2 will be able to connect to the server programs on AIX.

4.1.1 Installing and Configuring IBM COBOL Set for AIX

To port the Employee Lookup programs on the RS/6000 machine, you need to recompile and relink the COBOL source programs. You must have COBOL compiler installed on the AIX machine.

To install IBM COBOL Set for AIX, go through the following steps:

1. Log on as root.
2. Insert the CD-ROM into the right-hand drive and type the SMIT fast path command.
smitty install_latest
3. Select the drive where the CD-ROM is located and the required file sets to install these COBOL components.

In this section, **cobol.cmp.** and **cobol.rte** that are COBOL compiler and COBOL run-time system must be installed. As IBM COBOL Set for AIX

provides many excellent utilities, you are recommended to install these components of the COBOL product in addition to the compiler and run-time system.

Fastpath to install all the COBOL components.

If you intend to install all the components of IBM COBOL Set for AIX, you can enter the following command quick start the installation:

```
installp -ac -d device all
```

Here, *device* is the device name where the CD-ROM is located, for instance, /dev/cd0.

4. The COBOL runtime system requires the SOM modules. In order to execute COBOL programs, you also have to install the following SOM-related modules from the AIX system disk:

```
bos.som  
bos.dsom  
bos.som.rte  
bos.som.util
```

Now you have finished the installation of IBM COBOL Set for AIX. You can use command `cob2` to compile your COBOL programs to check whether the COBOL product works properly.

4.1.2 Installing and Configuring CICS for AIX Server

Before you start to install CICS for AIX, we recommend that you read *CICS for AIX Planning and Installation Guide*. The manual provides very detailed information regarding the installation and configuration of CICS for AIX. This section briefly describes one of the options that you can use to set up your CICS environment.

As the client/server environment you are to set up is mainly to demonstrate how the COBOL client/server applications successfully work in the CICS environment, the CICS environment described here is very simple and does not use the Distributed Computing Environment (DCE) Directory Service or DCE Security Service. It is the so-called non-DCE cell environment. In the real production environment, we recommend that your CICS environment use DCE Cell Directory Service and DCE Security Service for better control. In the sample application illustrated here, the CICS-for-AIX server environment uses the DB2-for-AIX-for-CICS queue and file management that is a new feature of CICS for AIX V 2.

If you need DCE Cell Directory Service and DCE Security Service, you must install more DCE components. If you need services from Encina, you also have to install more Encina components. Please refer to *CICS for AIX Planning and Installation Guide* for further information on DCE and Encina.

4.1.2.1 Installing DCE

CICS for AIX was developed with DCE application program interface (API) libraries and preprocessor. Therefore, DCE services are prerequisites for CICS for AIX. In the non-DCE cell environment, CICS for AIX requires only DCE remote procedure call (RPC). To install CICS components, you need to install the required DCE components first. Go through the following steps to install these components.

1. Log on as root.

2. Insert the AIX CD-ROM into the correct drive.

3. Type the following SMIT command:

smitty install_latest

4. Select the drive you are using and the following file sets:

- dce.client
 - dce.client.core.rte.admin
 - dce.client.core.rte.cds
 - dce.client.core.rte.config
 - dce.client.core.rte.rpc
 - dce.client.core.rte
 - dce.client.core.rte.time
 - dce.client.core.rte.zones
- dce.compat
 - dce.compat.cds.smit
 - dce.compat.client.core.smit
- dce.pthreads
 - dce.threads.rte
- dce.tools
 - dce.tools.admin.rte
 - dce.tools.appdev.rte

After the installation of DCE is finished, use **lspp -L "dce*"** to verify that you have installed all the required components.

If the required DCE file sets are all installed, go to next step to create the user and the groups required by CICS for AIX.

DCE RPC

DCE RPC can automate the development and implementation of all the communications associated with distributed applications. CICS for AIX uses DCE RPC to open connections, to organize data into a stream for transmission, and to transmit the data. RPC authentication allows you to place additional data protection and authentication on data transmitted between clients and servers. DCE RPCs provide several levels of security. They also provide endpoint mapping. When a server used by CICS is initialized, it writes its interface name and server process address, the end point, to an end-point map database. On each host with servers, an RPC daemon process uses the end-point map database to help clients identify servers. Each end-point map database contains information for all servers on that host.

4.1.2.2 Preparing Operating System for CICS

CICS for AIX requires the following user and groups.

- AIX group: **cics** and **cicsterm**
- AIX user: **cics** which is a member of the **cics** group.

CICS for AIX also requires the user root as a member of the **cics**.

The installation process of CICS for AIX does not create the users and groups for you. You have to create a user, CICS, and two groups, CICS, and CICS term, before you install CICS for AIX. Follow these steps to create them:

1. Log on as root.
2. Type the following command.
smitty mkgroup
Add a Group panel will appear.
3. Type **cics** in the Group NAME field and press Enter to add this group (Figure 200).

Add a Group

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

* Group NAME

ADMINISTRATIVE group?

Group ID

USER list

ADMINISTRATOR list

[Entry Fields]

[cics]

false

[]

[]

[]

F1=Help

F2=Refresh

F3=Cancel

F4=List

F5=Reset

F6=Command

F7=Edit

F8=Image

F9=Shell

F10=Exit

Enter=Do

Figure 200. Add a Group Panel

4. Repeat the previous two steps to create a group, **cicsterm**.
5. Type the following command to create an AIX user, CICS:
smitty mkuser
Add a User panel will be displayed.
6. Type **cics** in the User NAME field.
Type **cics** in the PRIMARY Group field.
Type **cicsterm** in the Group set field (Figure 201 on page 136).

Add a User			
Type or select values in entry fields. Press Enter AFTER making all desired changes.			
[TOP]	[Entry Fields]		
* User NAME	[cics]		
User ID	[]		
ADMINISTRATIVE USER?	false		
Primary GROUP	[cics]		
Group SET	[cicsterm]		
ADMINISTRATIVE GROUPS	[]		
Another user can SU TO USER?	true		
SU GROUPS	[ALL]		
HOME directory	[]		
Initial PROGRAM	[]		
User INFORMATION	[]		
EXPIRATION date (MMDDhhmmyy)	[0]		
Is this user ACCOUNT LOCKED?	false		
[MORE...31]			
F1=Help	F2=Refresh	F3=Cancel	F4=List
F5=Reset	F6=Command	F7=Edit	F8=Image
F9=Shell	F10=Exit	Enter=Do	

Figure 201. Add a User Panel

7. Make root a member of the CICS group by typing the following command:

smitty chuser

Add cics and cicsterm to the Group SET field and press Enter (Figure 202 on page 137).

Change / Show Characteristics of a User			
Type or select values in entry fields. Press Enter AFTER making all desired changes.			
[TOP]	[Entry Fields]		
* User NAME	root		
User ID	[0]		
ADMINISTRATIVE USER?	true		
Primary GROUP	[system]		
Group SET	<,cscbgrp,cics,cicsterm]		
ADMINISTRATIVE GROUPS	[]		
Another user can SU TO USER?	true		
SU GROUPS	[ALL]		
HOME directory	[/]		
Initial PROGRAM	[/bin/ksh]		
User INFORMATION	[]		
EXPIRATION date (MMDDhhmmyy)	[0]		
Is this user ACCOUNT LOCKED?	false		
[MORE...31]			
F1=Help	F2=Refresh	F3=Cancel	F4=List
F5=Reset	F6=Command	F7=Edit	F8=Image
F9=Shell	F10=Exit	Enter=Do	

Figure 202. Change/Show User Characteristics of a User—Root

- As you will run several CICS executable programs in the future, you should change the environment variable PATH. Modify PATH in the /etc/environment file so that all the users will also have this PATH setting.

Edit /etc/environment file by adding the following statement:

```
export PATH=$PATH:/usr/lpp/cics/bin
```

After all the required components and settings are set, you can start to install CICS for AIX.

4.1.2.3 Installing CICS for AIX

To install CICS for AIX, follow these steps:

- Log on as root.
- Insert the AIX media into the correct drive.
- Type the following SMIT command:

```
smitty install_latest
```

Select the drive where the AIX CD-ROM is located and the file sets listed below and press Enter to install them.

- cics.base
 - cics.base.adt
 - cics.base.rte
- cics.server

- cics.server.adt
- cics.server.adt.ivp
- cics.server.rte
- cics.server.rte.ivp
- cics.client
 - cics.client.adt
 - cics.client.rte
- cics.msg.language
 - cics.msg.en_US.base
- cics.info.en_US
- encina.client
- encina.server

The installation of CICS should be finished. You can verify the installed file sets using the following command:

```
smitty lslpp_installed
```

The command lists all the installed software file sets on the machine.

4.1.2.4 Configuring DCE

After the CICS for AIX is installed, you need to configure DCE. In the non-DCE-cell environment, use the following command to configure DCE:

```
mkdce -n cellname rpc
```

This command requires an entry as the cell name. You must type something as *cellname* even if you do not expect to use the DCE CELL Directory service.

To configure a DCE client for a CICS client or region in the non-DCE-cell environment, type the following command:

```
cicssetupclients -m
```

Use this command once for each server machine before you run a region or terminal. The DCE configuration is completed with these steps.

4.1.2.5 Configuring DB2 for Queue and File Management

CICS queue and file management consists of the management of data associated with user files, auxiliary temporary storage queues, transient data queues, and locally queued, automatic, transaction-initiation requests. The data can be managed either with an Encina SFS or with DB2 for AIX V2. To make the environment simple, DB2 for AIX is used for the queue and file management in this section.

You have DB2 for AIX installed on your RS/6000 machine, so you need only take the following steps to configure DB2 for queue and file management:

1. Log on as instance owner—in this example, cscbadm.
2. Create a database used for queue and file management,

```
db2 create database databasename
```

where *databasename* is the name of the database that will store the files and queues.

In this example, csdbdb.

3. Type **db2start** to start the database.

To enable the interface between CICS for AIX and DB2 for AIX, go through the following steps.

1. Log on as root.
2. As you have created a symbolic link from the system library in Chapter 1, “OS/2 Local Area Network with DB2” on page 1, you should now add another symbolic link:

```
ln -s /usr/lpp/db2_02_01/lib/db2.o /usr/lib/db2.o
```

Symbolic links allow your programs to be linked without reference to a specific release of DB2. If you do not have these symbolic links, you will have to relink your programs when you update to a newer release.

3. The CICS COBOL run-time, CICS transactions, and the switch load file all need to reference the DB2 shared object at run-time. Enter these commands to create a DB2 for AIX shared object:

```
cd /usr/lpp/db2_02_01/lib
ar -vx libdb2.a
mv shr.o db2.o
```

Note

The DB2 shared object db2.o must be used as input when linking IBM COBOL programs. This ensures that both CICS and the CICS programs share the same copy of DB2.

4. Give the user, CICS, DBA authority to have appropriate database access privileges. Type

smitty chuser

Add the DB2 administration group, in this example, cscbgrp, to the Group SET field and press Enter (Figure 203 on page 140).

Change / Show Characteristics of a User	
Type or select values in entry fields. Press Enter AFTER making all desired changes.	
[TOP]	[Entry Fields]
* User NAME	cics
User ID	[201]
ADMINISTRATIVE USER?	false
Primary GROUP	[cics]
Group SET	<cics,cicsterm,cscbgrp]
ADMINISTRATIVE GROUPS	[]
Another user can SU TO USER?	true
SU GROUPS	[ALL]
HOME directory	[/home/cics]
Initial PROGRAM	[/usr/bin/ksh]
User INFORMATION	[]
EXPIRATION date (MMDDhhmmyy)	[0]
Is this user ACCOUNT LOCKED?	false
[MORE...31]	
F1=Help	F2=Refresh
F5=Reset	F6=Command
F9=Shell	F10=Exit
F3=Cancel	F4=List
F7=Edit	F8=Image
Enter=Do	

Figure 203. Change/Show User Characteristics of a User—CICS

Why grant authority to CICS?

At region startup, the CICS application server makes an initial connection to DB2 using the XA open string. By default, CICS connects to the DB2 databases that were set up with CLIENT authentication, as user CICS. All CICS transactions run with the database privileges of this user. Therefore, the DB2 system administrator must grant appropriate privileges on the appropriate databases to this user ID.

Now you have completed the configuration of DB2 for queue and file management in the CICS for AIX environment.

4.1.2.6 Setting up CICS Region

A region is a collection of CICS-controlled resources as a unit. A region includes programs, BMS map sets, transactions, terminals, files, transient data queues, temporary storage queues, journals, products, and users. Multiple regions can exist simultaneously one CICS for AIX system. It is likely that CICS regions are application-specific. From the users' point of view, each CICS region is like an independent CICS system.

To use CICS for AIX, you need to have at least one region created and started. Then you will be able to work with CICS for AIX.

To create a CICS region, follow these steps:

1. If you use DB2 for the queue and file management, make sure the database has been started by the database instance owner.

2. Log on as root.

3. Type the following command to create a region:

```
cicscp create region regionname -oinstancename -adatabasename
```

regionname is the name of the region that will be created. *instancename* is the database instance name. *databasename* is the database used for CICS file and queue management.

In this example, we use **cscbrgn** as *regionname*, **cscbadm** as *instancename* and **cscbdb** as *databasename*

This command will start DCE RPC processes, set up the database as the CICS file server, and name the instance owner.

To create and define a user to CICS, follow these steps:

1. Enter the SMIT command:

```
smitty cicsaddud
```

2. You are prompted to select a model as the base definition of the new user. Press Enter to select "" as the default model.

3. Enter the user name in the User Identifier field.

4. Set **none** in the DCE Security group field.

5. Press Enter to add the user. (Figure 204 on page 142)

Add User															
Type or select values in entry fields. Press Enter AFTER making all desired changes.															
	[Entry Fields]														
* User Identifier	[CSCBU1]														
* Model User Identifier	"														
* Region name	[cscbrgn]														
Add to database only OR Add and Install	Add AND Install														
Ignore errors?	no														
Group to which resource belongs	[]														
Activate the resource at cold start?	yes														
Resource description	[User Definition]														
* Number of updates	0														
Protect resource from modification?	no														
Transaction Level Security Key List	[1]														
Resource Level Security Key List	[none]														
DCE principal of the user	[]														
User priority	[0]														
User Trace filename	[]														
Operator ID	[AIX]														
(Obsolete) DCE cell name	[]														
Encrypted password	[]														
* Add or change user in DCE Security Registry?	yes														
* DCE Security group	[none]														
* DCE Security organization	[none]														
* HOME directory	[/]														
* Initial PROGRAM	[cicsterm]														
<table border="0"> <tr> <td>F1=Help</td> <td>F2=Refresh</td> <td>F3=Cancel</td> <td>F4=List</td> </tr> <tr> <td>F5=Reset</td> <td>F6=Command</td> <td>F7=Edit</td> <td>F8=Image</td> </tr> <tr> <td>F9=Shell</td> <td>F10=Exit</td> <td>Enter=Do</td> <td></td> </tr> </table>				F1=Help	F2=Refresh	F3=Cancel	F4=List	F5=Reset	F6=Command	F7=Edit	F8=Image	F9=Shell	F10=Exit	Enter=Do	
F1=Help	F2=Refresh	F3=Cancel	F4=List												
F5=Reset	F6=Command	F7=Edit	F8=Image												
F9=Shell	F10=Exit	Enter=Do													

Figure 204. Add User—CICS

Now you are ready to start the CICS region.

Type the following command to start the region that you just created.

```
cicscp start region regionname
```

where *regionname* is the name of the region.

You can check the log file, console.msg, by entering:

```
tail -f /var/cics_regions/cscbrgn/console.msg
```

When you see the message '***CICS for AIX startup is complete***', in the console.msg file, the region is successfully started. If you do not see this message, trace the log file to find the clues needed to resolve the issue.

4.1.2.7 Configuring CICS for AIX Server

In Chapter 3, “COBOL with DB2 for AIX and DB2 Client” on page 109, you should have finished the TCP/IP configuration of the server machine. In order to communicate with the CICS clients, you need to define and configure a Listener in the region you are working with.

Listener

CICS for AIX supports requests from the IBM CICS Client product by using a listener process. This process is started at region startup if a Listener Definitions (LD) entry exists with the protocol attribute set to TCP. The listener process communicates with a corresponding process on the client machine. The communication between these processes imitates the flows between CICS regions. The flows generated by an ECI request are the same as those for a CICS-to-CICS distributed program link (DPL) request. The client process appears to the server as a limited terminal-owning region.

To define a listener for the region, go through these steps.

1. Enter the SMIT command:

```
smitty cicsaddld
```

2. A window labeled Define Listener appears, Figure 206 on page 145.

When you are prompted to select a Model Listener Identifier, press Enter to select the default listener definition that will be displayed in the window. You then enter the required information and leave all the remaining values as default. Enter a name, for instance, CSCBLD, in the Listener Identifier entry field.

In the entry field for Add to database only or Add and Install, if you select Add and Install, the definition of the listener will be available to the CICS clients immediately. If you select Add, the definition will not be available until you shut down the region and restart it.

3. Type the IP address or the host name of the server machine in the TCP adapter address field.

Note

If you have not applied CICS PTF Level 4 or above on your CICS for AIX, and you type the IP address in the TCP adapter address, you get an error message when you restart the region. To avoid that, type the host name instead of the IP address. Ensure that the host name is defined in the name server or /etc/hosts file. It is highly recommended that you apply the latest CICS PTFs.

4. Type service name in the TCP service name. The reserved CICS port is 1435/tcp; if you use the default port, you can leave it blank, but you have to ensure that no other TCP/IP service uses the same port.

If you use another port, enter the service name of the port and ensure that you have added the port definition to the '/etc/services' file. The port number has to be unique.

5. Press Enter to add the listener definition.
6. To activate the listener, shut down the CICS region by using the command:

```
cicscp stop region regionname
```

and restart it using the start region command again.

```
cicscp start region cscbrgn
```

As described before, you can check the log file, 'console.msg'. If you see the message '***CICS for AIX startup is complete***', in the file, the region is successfully started and ready to communicate with CICS clients. Also, you can check the messages related to the listener definition to ensure that the listener process has started properly (Figure 205).

```
...  
...  
... Creating CICS process 'cicsip' for listener 'CSCBLD'  
... Application manager now starting MinServer servers  
... Application server 8 started  
... CICS listener process 'cicsip' started for '9.112.34.63 [1435] '  
...  
...
```

Figure 205. AIX Console Message

If the region can not be started, trace the error messages in the log file to find the reason and resolve it. Mostly the errors occurred after the listener being defined are caused by the wrong TCP adapter address or the undefined TCP service name.

Add Listener													
Type or select values in entry fields. Press Enter AFTER making all desired changes.													
	[Entry Fields]												
* Listener Identifier	[CSCBLD]												
* Model Listener Identifier	" "												
* Region name	[cscbrgn]												
Add to database only OR Add and Install	Add AND Install												
Group to which resource belongs	[]												
Activate resource at cold start?	yes												
Resource description	[Listener Definition]												
* Number of updates	0												
Protect resource from modification?	no												
Protocol type	TCP												
TCP adapter address	[cscbaix.stl.ibm.com]												
TCP service name	[]												
local SNA Server Protocol Type	TCP												
local SNA Server Identifier	[]												
local SNA Node Name	[]												
<table> <tr> <td>F1=Help</td> <td>F2=Refresh</td> <td>F3=Cancel</td> <td>F4=List</td> </tr> <tr> <td>F5=Reset</td> <td>F6=Command</td> <td>F7=Edit</td> <td>F8=Image</td> </tr> <tr> <td>F9=Shell</td> <td>F10=Exit</td> <td>Enter=Do</td> <td></td> </tr> </table>		F1=Help	F2=Refresh	F3=Cancel	F4=List	F5=Reset	F6=Command	F7=Edit	F8=Image	F9=Shell	F10=Exit	Enter=Do	
F1=Help	F2=Refresh	F3=Cancel	F4=List										
F5=Reset	F6=Command	F7=Edit	F8=Image										
F9=Shell	F10=Exit	Enter=Do											

Figure 206. Add Listener Panel—Define Listener

4.1.3 Compiling and Linking the CICS Server Program

The CICS COBOL programs used here are in the diskette accompanying this redbook. The sources are also listed in the Appendix.

The program will access the sample database created by the instance owner, cscbadm, to make the environment simple. When you work on the sample programs in this section, log on as the instance owner, cscbadm.

To copy the source files from the diskette, make a directory and change to the directory. Enter this command:

```
tar -xvf /dev/fd0
```

All the required files will be copied to the server machine. The source programs are the same as those used in Chapter 3, "COBOL with DB2 for AIX and DB2 Client" on page 109. A file named empllu.mk is the make file to translate, compile, and link the server programs. The make file is listed below (Figure 207 on page 146):

```

all: emplusc.o emplusv.ibmcb

emplusv.ibmcb: emplusv.cbl
cob2_r -v -qLIB -bI:/usr/lpp/cics/lib/cicsprIBMCOB.exp \
-e _iwz_cobol_main -L/usr/lib/dce -ldcelibc_r \
-ldcepthreads -I/usr/lpp/cics/include \
-I/usr/lpp/db2_02_01/include/cobol_a /usr/lib/db2.o \
-qQUOTE -g -o emplusv.ibmcb emplusv.cbl emplusc.o

emplusv.cbl: emplusv.ccp
cicstran -e -d -lIBMCOB -qQUOTE emplusv.ccp

emplusc.o: emplusc.cbl
cob2 -c -g -qlib emplusc.cbl

```

Figure 207. EMPLLU Makefile

The name of the server program is `emplusv.ccp`, which differs from the name used in previous chapters, because the 'cicstran' command requires the extension of program name as 'ccp'.

The reason we coded 'cob2' command rather than use 'cicstcl' is that `emplusv.ccp` calls a subroutine, `emplusc`, which is a non-CICS program. As the main entry point of the entire server application should be `emplusv`, when the programs are linked, the linker has to recognize the main entry is `emplusv` rather than `emplusc`. If you use `cicstcl` command, first you have to compile `emplusc.cbl` to create `emplusc.o` using 'cob2' command. Then you set an option to link `emplusc.o` module to the `cicstcl` command. However, the `cicstcl` will interpret `emplusc` as the main entry point that will cause problem. You will get some error messages such as exception error when the CICS program just started, because the entry point is incorrect. To work around this restriction, write the `cob2` command manually as the make file illustrated here.

This make file is used by COBOL Set for AIX V1.2 that consists of DB2 Coprocessor. Therefore, you do not have to precompile the COBOL program with SQL statement. However, if you use COBOL Set for AIX V1.1, you have to add the following statements in the make file to precompile the COBOL programs:

```

emplusv.ccp: emplusv.sqb
    db2 connect to sample;
    db2 prep emplusv.sqb target ibmcb; \
    db2 grant execute on package emplusv to public; \
    mv emplusv.cbl emplusv.ccp

```

To compile and link the programs, ensure that the database has been started. Then type

make -f emplus.mk

You will get 'emplusv.ibmcb' file in the current directory. It is the CICS server execution program and you have to define it in the CICS resource table that the client program can invoke it through ECI call.

4.1.4 Defining CICS Resources for the CICS Server Program

In CICS for AIX, you have to use 'smit' to define CICS resources. It is different from other CICS products. CICS for AIX uses region- related storage definitions for resource storage, instead of using xxT tables.

To define CICS resources, enter the SMIT command:

smitty cicsresource

The panel will appear as shown in Figure 208. You can define all the CICS resources from this panel as well as other processes required by CICS for AIX. You can also use SMIT fast paths to go through a different resource definition panel directly. In the remaining sections of the chapter, SMIT fast paths will be used so that you can go to the desired resource definition panel directly.

Manage Resource(s)

Move cursor to desired item and press Enter.

Communications
Files
Gateways
Journals
Listeners
Monitoring
Programs
Region
Temporary Storage Queues
Terminals
Transactions
Transient Data Queues
Users
XA Definitions

F1=Help
F9=Shell

F2=Refresh
F10=Exit

F3=Cancel
Enter=Do

F8=Image

Figure 208. Manage Resource(s) Panel—Define CICS Resources

To define the CICS execution program that you just built, you can either use the previous panel to select the resources you define, or type the SMIT fastpaths:

smitty cicsaddprogram

You will see the following panel (Figure 209 on page 148):

- Type the program identifier, **EMPLLUSV** in the entry field.
- Type the path of the CICS execution program.
- Define the Resource Level as **public** that means all users can execute the program.
- Define the type as **program**.

Shutdown the CICS region and restart it. The server program should be available for client programs to use.

Add Program													
Type or select values in entry fields. Press Enter AFTER making all desired changes.													
	[Entry Fields]												
* Program Identifier	[EMPLLUSV]												
* Model Program Identifier	" "												
* Region name	[cscbrgn]												
Add to database only OR Add and Install	Add AND Install												
Group to which resource belongs	[]												
Activate resource at cold start?	yes												
Resource description	[Program Definition]												
* Number of updates	0												
Protect resource from modifications?	no												
Program enable status	enabled												
Remote system on which to run program	[]												
Name to use for program on remote system	[]												
Transaction name on remote system for program	[]												
Resource Level Security Key	[public]												
Program path name	<regions/cscbrgn/EMPLLUSV]												
Program type	[program]												
User Exit number	[0]												
Is a user conversion template defined?	no												
Is this a program that should be cached?	no												
<table> <tr> <td>F1=Help</td> <td>F2=Refresh</td> <td>F3=Cancel</td> <td>F4=List</td> </tr> <tr> <td>F5=Reset</td> <td>F6=Command</td> <td>F7=Edit</td> <td>F8=Image</td> </tr> <tr> <td>F9=Shell</td> <td>F10=Exit</td> <td>Enter=Do</td> <td></td> </tr> </table>		F1=Help	F2=Refresh	F3=Cancel	F4=List	F5=Reset	F6=Command	F7=Edit	F8=Image	F9=Shell	F10=Exit	Enter=Do	
F1=Help	F2=Refresh	F3=Cancel	F4=List										
F5=Reset	F6=Command	F7=Edit	F8=Image										
F9=Shell	F10=Exit	Enter=Do											

Figure 209. Add Program Panel—Define CICS Program

As the CICS server program, empllusv, consists of SQL statements to access the data in the sample database, you have to enable the XA interface between CICS for AIX and DB2. The following steps must be done: (As you have done Step 1 to Step 4 when you configure DB2 as CICS file and queue management, you can skip Steps 1 to 4).

1. Create the symbolic links:
 - execute db2ln
 - `ln -s /usr/lpp/db2_02_01/lib/db2.o /usr/lib/db2.o`
2. Create a DB2 shared object:
 - `cd /usr/lpp/db2_02_01/lib`
 - `ar -vx libdb2.a`
 - `mv shr.o db2.o`
3. Grant CICS database access privileges:

- Grant bindadd on database to user CICS
- Grant select on table system.sysindexes to user CICS

4. Set the \$DBINSTANCE

Add DB2INSTANCE=cscbadm to /var/cics_regions/cscbrgn/environment file

5. Create the Product Definition (XAD)

XAD is used to define transactional products that use the X/Open XA protocol. Each XAD entry contains information for one product. To add an XA definition, enter the following SMIT command and XA Definition window will show up. Enter the required data to the proper entry fields. You have to specify the XAD identifier, SwitchLoadFile and Resource Manager Termination String (XAOpen).

XAOpen defines the XA open string which is passed to the DB/2's xa_open_entry() function call. For DB2, the string consists of:

database[, username, password]

In our example, the XAOpen string is the name of the database, sample, that will be used in the EMPLLU application.

SwitchLoadFile should be set as a path name to an object file that contains the xa_switch_t structure definition and XA support subroutines for DB2. The string should be setup to point to \$CICS/bin/cicsxadb2, the fully qualified path name of the supplied cicsxadb2 object file. Figure 210 shows the sample XA definition used in this example.

Add XA Definition			
Type or select values in entry fields. Press Enter AFTER making all desired changes.			
* XA Definition Identifier	[Entry Fields] [samplexa]		
* Model XA Definition Identifier	"		
* Region name	[cscbrgn]		
Add to database only or Add and Install	Add AND Install		
Group to which resource belongs	[]		
Activate resource at cold start?	yes		
Resource description	[XA Product Definition]		
* Number of updates	0		
Protect resource from modification?	no		
Switch Load File Path Name	[/usr/lpp/cics/bin/cics>		
Resource Manager Initialization String	[SAMPLE]		
Resource Manager Termination String	[]		
Resource Manager Serialization Attribute	all_operations		
F1=Help	F2=Refresh	F3=Cancel	F4=List
F5=Reset	F6=Command	F7=Edit	F8=Image
F9=Shell	F10=Exit	Enter=Do	

Figure 210. Add XA Definition Panel

Relational database support

CICS transactions/programs can access RDBMs by including embedded SQL calls within the body of CICS programs. Coordinated commitment and the recovery of transactions that include both CICS and SQL calls is possible only with RDBMs that support the X/Open XA interface, as defined by the X/Open Distributed Transaction Processing standard. In the CICS for AIX implementation of the X/Open DTP standard, the transaction manager (CICS) coordinates transaction initiation and completion among the resource managers (RDBMs) as initiated by a CICS application.

The CICS RDBM transaction support allows application programmers to use the CICS API to do the following when accessing data in a RDBM:

- Start global transactions that the applications may request from the RDBMs with the transaction.
- Roll back transactions in the event of a failure in the first phase of a two-phase commit.
- In the case of DB2, single-phase commit support is provided, giving improved performance.
- Restore shared resources to a consistent state after a failure.
- DB2 can use the CICS RDBM transaction support provided with CICS for AIX.

4.2 CICS Client for OS/2 Environment Setup

This section describes how to configure your CICS Client for OS/2 environment that your client COBOL program can use ECI calls to communicate with the CICS server program on CICS for AIX environment.

4.2.1 Configuring CICS Client to Connect to CICS for AIX

In order to access the CICS for AIX server machine, you have to modify the initialization file for the CICS client. The default initialization file name is CICSCLI.INI.

Type

```
cd d:\cicscli\bin
```

(d: is the drive where the CICS Client for OS/2 is installed).

Edit cicscli.ini file and go to the Server section.

Modify the following information based on your CICS server:

```
Server = CICSTCP
```

```
Protocol = TCPIP
```

```
NetName = hostname
```

```
Port = portname
```

TCP/IP is the protocol that supports CICS for AIX—CICS Client for OS/2 environment. You have to define the server with TCP/IP protocol. NetName is the host name or IP address of the CICS server. If you use the default port, 1435, set

the Port value as 0; otherwise, you have to specify the port name as the value of Port.

In the initialization file, you have to define the driver name that will be used with the communication protocol. Ensure the following statements existing in the file to specify the driver.

```
Driver = TCP/IP  
DriverName = CCLIBMIP
```

Figure 211 shows an example of the initialization file.

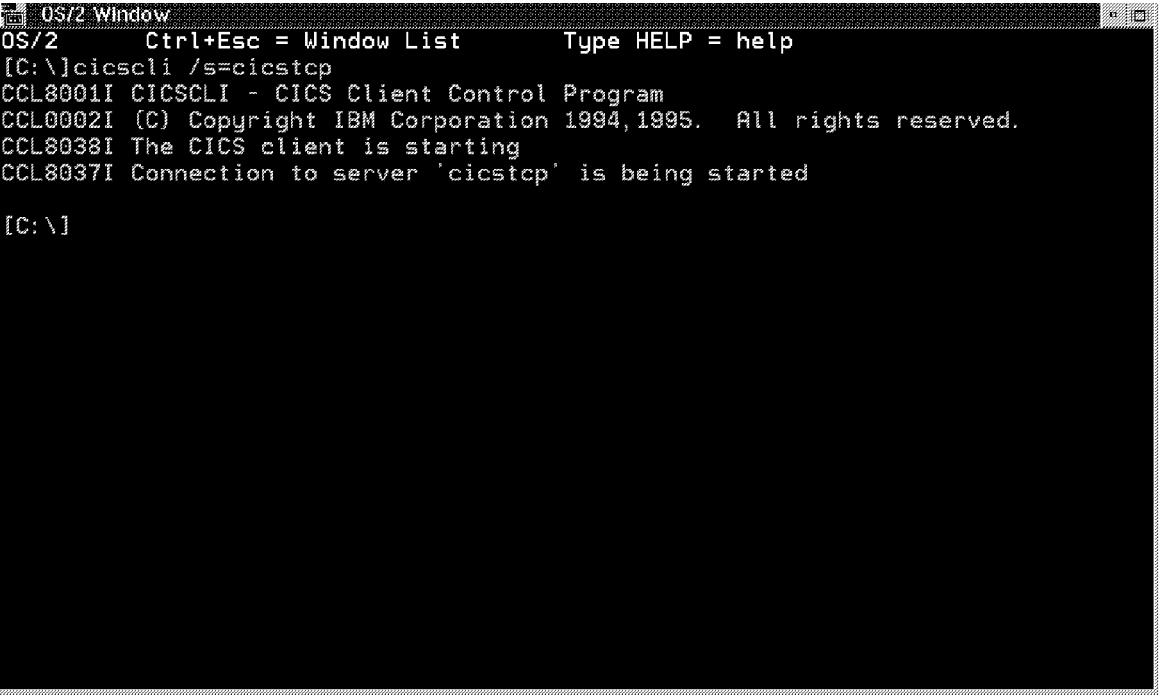
```
Client = *  
  MaxServers = 1  
  MaxRequests = 20  
  MaxBufferSize = 32  
  LogFile = CICSCLI.LOG  
  TraceFile = CICSCLI.TRC  
  DosMemory = 48  
  
Server = CICSTCP  
  Description = TCP/IP Server  
  Protocol = TCPIP  
  Port = 0  
  NetName = cscbaix.stl.ibm.com  
  
Driver = TCPIP  
  DriverName = CCLIBMIP
```

Figure 211. CICSCLI.INI Initialization File

Type the following command to start the CICS client with the CICS for AIX server:

```
cicscli /s=cicstcp
```

The result is a message informing you that the connection is ready (Figure 212 on page 152).

A screenshot of an OS/2 window titled "OS/2 Window". The window contains a command prompt interface. At the top, it shows "OS/2 Ctrl+Esc = Window List Type HELP = help". Below this, the command "[C:\]cicscli /s=cicstop" has been entered. The output consists of several lines of text: "CCL8001I CICSCLI - CICS Client Control Program", "CCL0002I (C) Copyright IBM Corporation 1994,1995. All rights reserved.", "CCL8038I The CICS client is starting", and "CCL8037I Connection to server 'cicstop' is being started". The prompt "[C:\]" is visible at the bottom of the text area.

```
OS/2      Ctrl+Esc = Window List      Type HELP = help
[C:\]cicscli /s=cicstop
CCL8001I CICSCLI - CICS Client Control Program
CCL0002I (C) Copyright IBM Corporation 1994,1995. All rights reserved.
CCL8038I The CICS client is starting
CCL8037I Connection to server 'cicstop' is being started

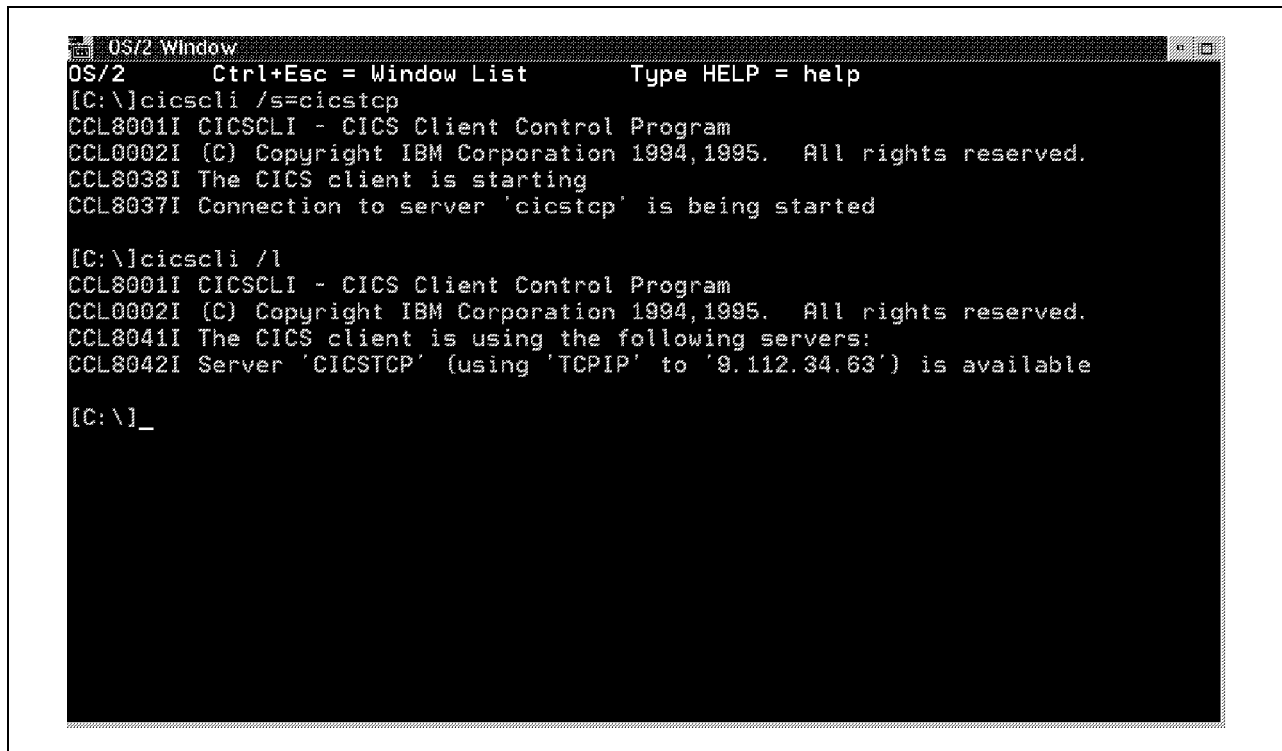
[C:\]
```

Figure 212. OS/2 Window—Start CICS Client

You can check the connection status using the command:

```
cicscli /l
```

The information you will get from this command is shown in Figure 213 on page 153.

The image is a screenshot of an OS/2 terminal window. The title bar at the top reads "OS/2 Window". Inside the window, the following text is displayed:
OS/2 Ctrl+Esc = Window List Type HELP = help
[C:\]cicscli /s=cicstop
CCL8001I CICSCLI - CICS Client Control Program
CCL0002I (C) Copyright IBM Corporation 1994,1995. All rights reserved.
CCL8038I The CICS client is starting
CCL8037I Connection to server 'cicstop' is being started

[C:\]cicscli /l
CCL8001I CICSCLI - CICS Client Control Program
CCL0002I (C) Copyright IBM Corporation 1994,1995. All rights reserved.
CCL8041I The CICS client is using the following servers:
CCL8042I Server 'CICSTCP' (using 'TCPIP' to '9.112.34.63') is available

[C:\]_

Figure 213. OS/2 Window—Check CICS Client

If you get the message informing you that the server is available, the CICS for AIX - CICS for OS/2 Client environment should be configured completely. You can use **cicsterm** to test if your CICS client can be running as a terminal. You can also try some predefined CICS transactions to see if the CICS client works properly. For instance, you can try the sign-on transaction, CESN, and the sign-on screen should appear as Figure 214 on page 154.

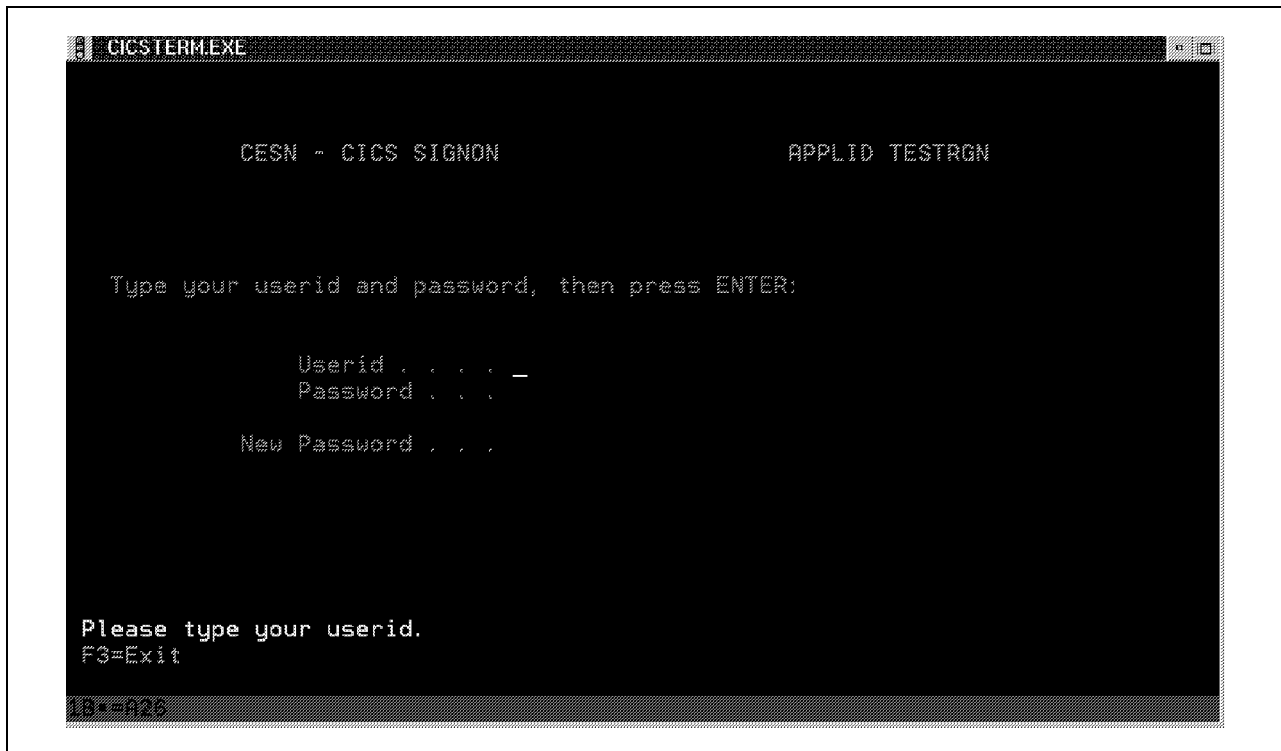


Figure 214. CICS TERM.EXE Window—CICS Sign-on

Note

If you want to have several CICS client initialization files to connect to different servers, specify the initialization file when you start the CICS client using the following command:

```
cicscli /f=filename
```

filename is the name of the initialization file.

Or you can set the environment variable, CICSCLI, to the name of the initialization file. For instance, command from the

```
set CICSCLI=cicscli.aix
```

When you start the CICS client, it will connect to the server defined in the cicscli.aix file.

4.2.2 Compiling and Linking the CICS Client Program

The client program is the same so you do not have to recompile and relink the program. Once you specify the server as the CICS for AIX server, your client program will connect to the server program on AIX. That means server programs can be ported to different platforms and still work with the same client programs. That is a very important feature and benefit of CICS products.

4.2.3 Executing the CICS Client/Server Programs

To execute the sample CICS client/server program, follow these steps:

1. Start TCP/IP on the server machine.
2. Start the CICS region, in this example, cscbrgn, on the server.
3. Start TCP/IP for OS/2 on the client.
4. Double-click on the EMPLLU icon of the desktop.
5. Press mouse button 2 on the EMPLLUCL project icon to run the CICS client program (Figure 215).
6. The result will be displayed on the list box. The result is stored in the database of the server machine accessed by the server program.

You have completed the CICS client/server programs in an environment of CICS for AIX server—CICS Client for OS/2. You also have learned the high portability of the CICS solution.

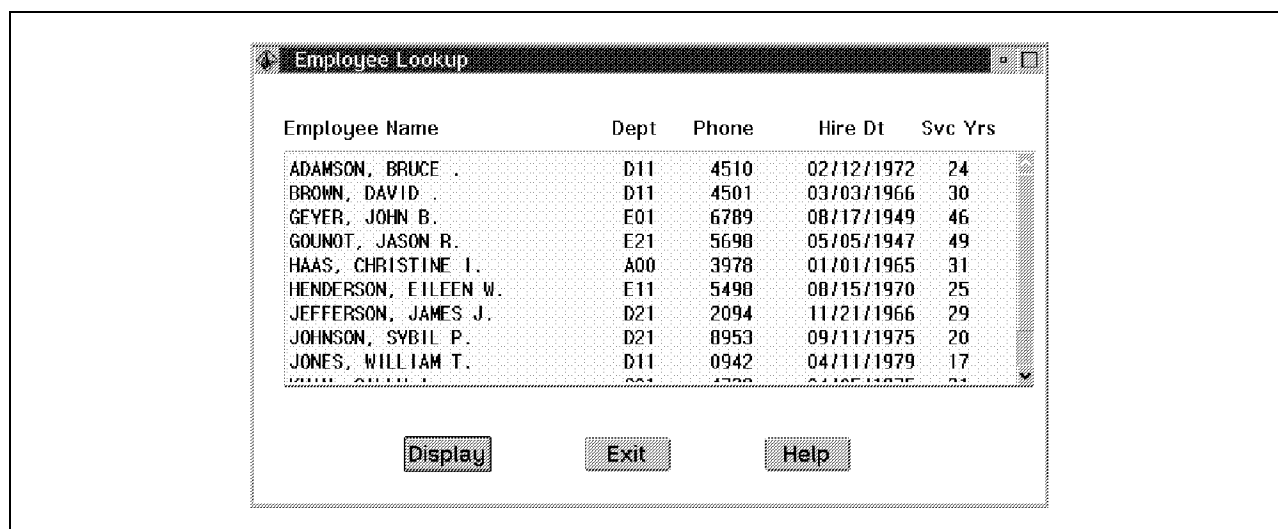


Figure 215. Employee Lookup Window—EMPLLU Run on the CICS Client

Chapter 5. Host Development Offload

There are several reasons why you would want to offload your development environment from a host system to a LAN environment:

- Make the host resources you would need for the development available to the production.
- Use the power of workstation editors and graphical user interfaces to enhance the programmer's productivity.
- Use powerful tools such as the debugger available under OS/2.
- Make a first step toward client/server computing by developing applications that also run on the workstation.

This chapter describes how to use the LAN development environment for the development or maintenance of a COBOL application which is targeted to run in an MVS environment.

The possible steps are these:

1. Download the existing source and test data from MVS to the workstation.
2. Set up the project on the workstation.
3. Change or enhance the application, build the project (preprocess, compile, link), and test the application using the debugger if necessary.
4. Transfer the updated source code back to the MVS system.
5. Prepare the application for running on the MVS system.

If you are developing a new application, you can omit the Step 1.

When you go through the example described on the following pages, you will not download any data from your MVS system. Instead, you will copy the source files from the diskette that comes with this book. Nevertheless, we describe in 5.2, "Download the Necessary Data to the Workstation" on page 168 how to download the data if you want to work on your own legacy applications.

The following versions of host products have been installed on the system we used for our example:

- MVS/ESA SP JES2 Version 5.2
- ISPF Version 4.2
- CICS for MVS/ESA Version 4.1
- DB2 for MVS/ESA Version 4.1
- IBM COBOL for MVS & VM Version 1.2.
- Language Environment for MVS & VM Version 1.5

IBM COBOL for MVS & VM Version 1.2. is almost totally source-compatible with IBM VisualAge for COBOL for OS/2. The few exceptions are very well documented in the COBOL Language Reference manual.

Figure 216 on page 158 shows an overview of the environment.

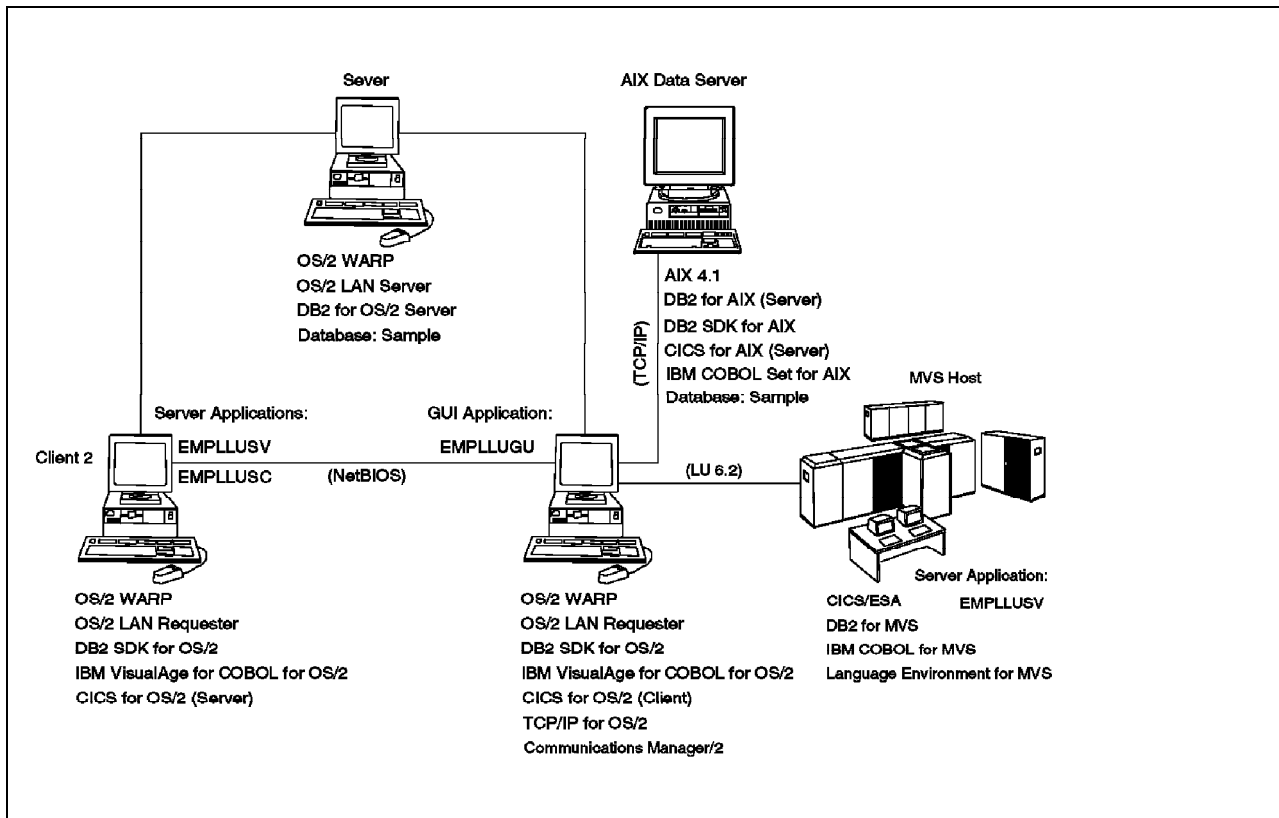


Figure 216. Host Development Offload

This environment is based on the setup described in Chapter 2, “OS/2 LAN with CICS and DB2” on page 59.

Workstation setup

The installation of the workstation products has been described in previous chapters. The only additional product needed for the environment shown in Figure 216 is Communications Manager/2 for OS/2 Warp.

Host environment

The installation and customization of the MVS products is not described in this book.

The panels and dataset names in your MVS environment might be different from the system we used in our example. Ask your system programmer about the naming conventions in your environment.

5.1 Installing and Configuring Communications Manager/2 for OS/2 Warp

For application development on the PC and the upload of the finished and tested programs to the mainframe you must establish a link between workstation and mainframe. You can then open a 3270 emulator session and upload the necessary files to the host.

First, you need to install Communications Manager/2 for OS/2 Warp on your workstation to establish an emulator session and to upload the files of the application to the host.

To install Communications Manager/2 for OS/2 Warp, go through the following steps:

1. Insert the Communications Manager CD-ROM in the CD drive. At an OS/2 command prompt, type

D:\CM2\CMSETUP

where D is the drive letter of the CD-ROM, and press Enter.

2. Click on the **OK** push button on the Installation of Communications Manager window (Figure 217).

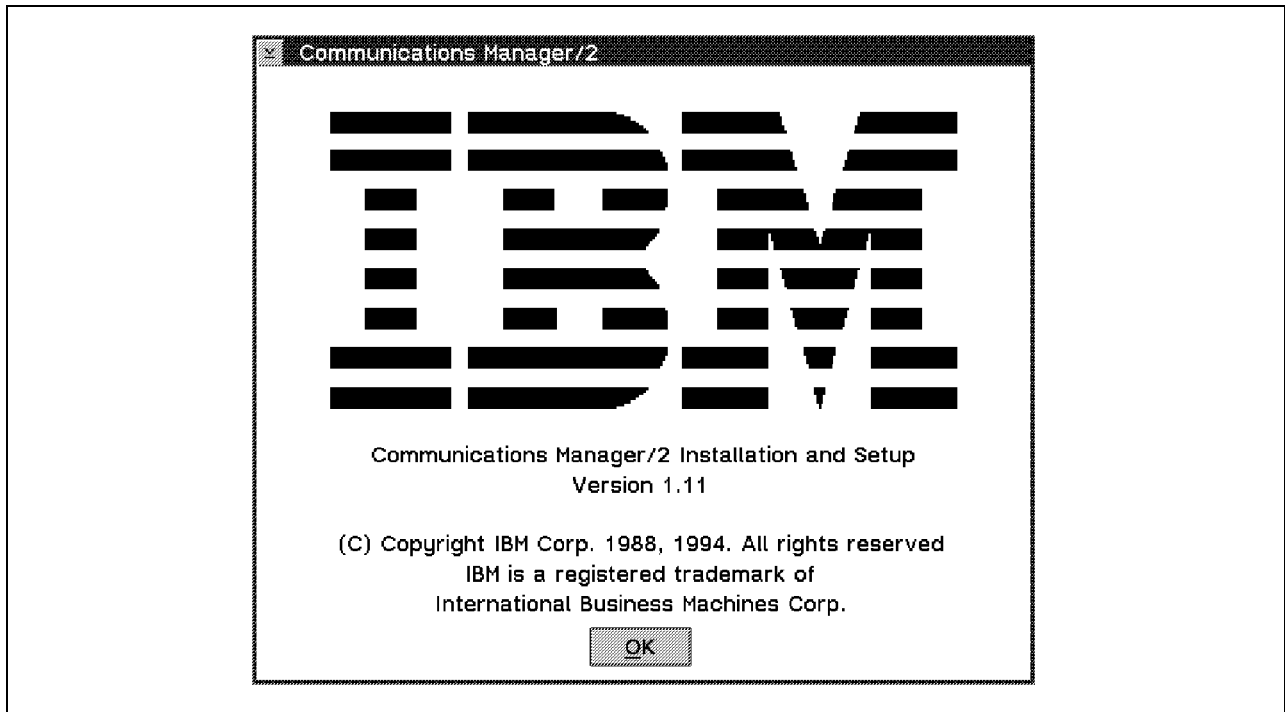


Figure 217. Installation of Communications Manager Window

3. The Installation Notes window appears. Click on **Continue**.
4. On the Target Drive Selection window (Figure 218 on page 160) select the drive where you want to install Communications Manager/2 for OS/2 Warp and click on **OK**.

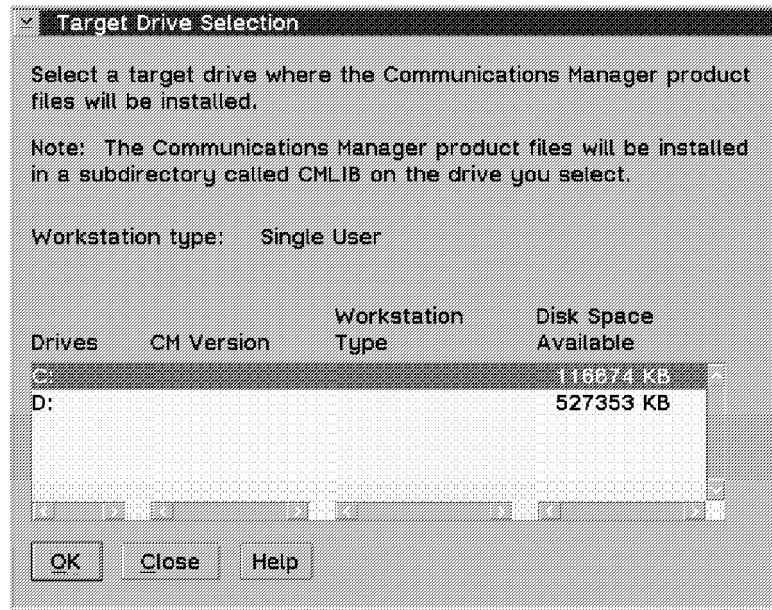


Figure 218. Target Drive Selection Window

5. On the Communications Manager Setup window (Figure 219) select **Setup...** to configure the machine.

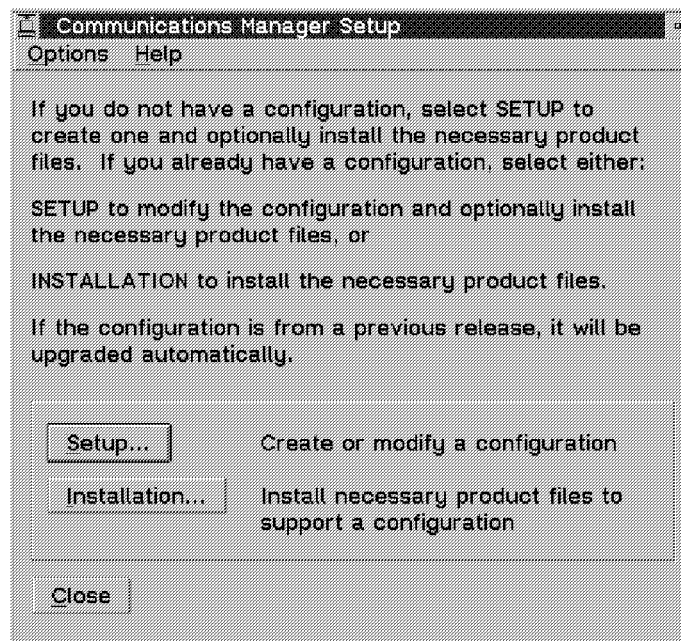


Figure 219. Communications Manager Setup Window

6. While the system copies files from the CD-ROM, it shows in the Copying Files window that it still works (Figure 220 on page 161).

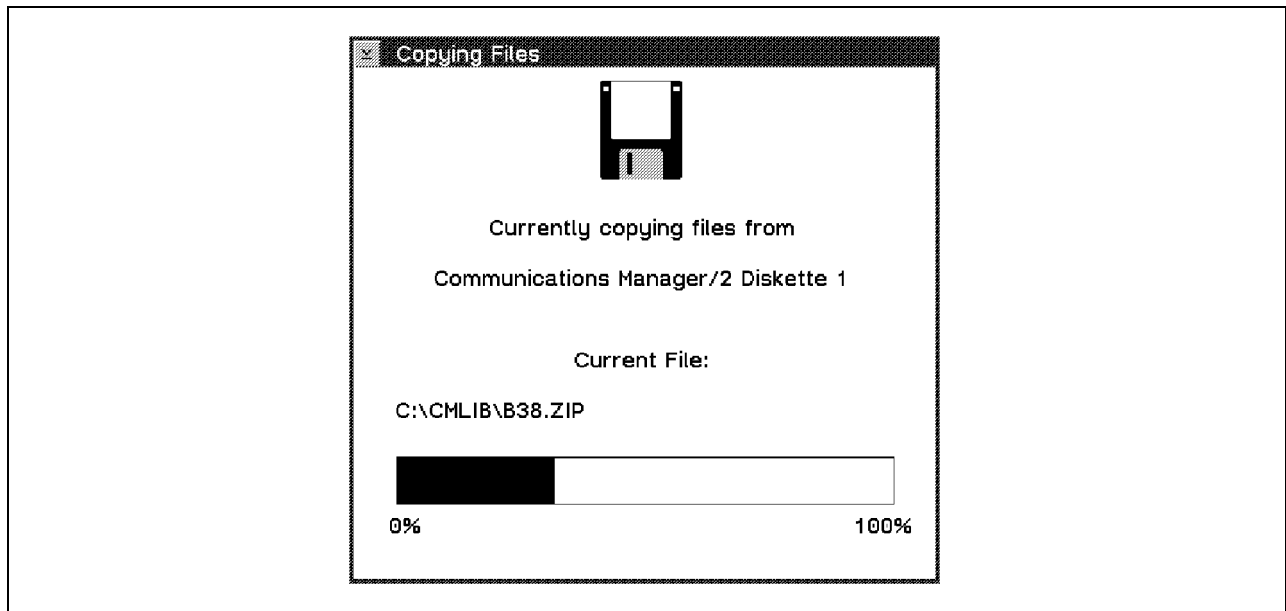


Figure 220. Copying Files Window

7. The Open Configurations window appears (Figure 221). Type a name for this configuration in the appropriate field (a description for the configuration is optional) and click on **OK**.

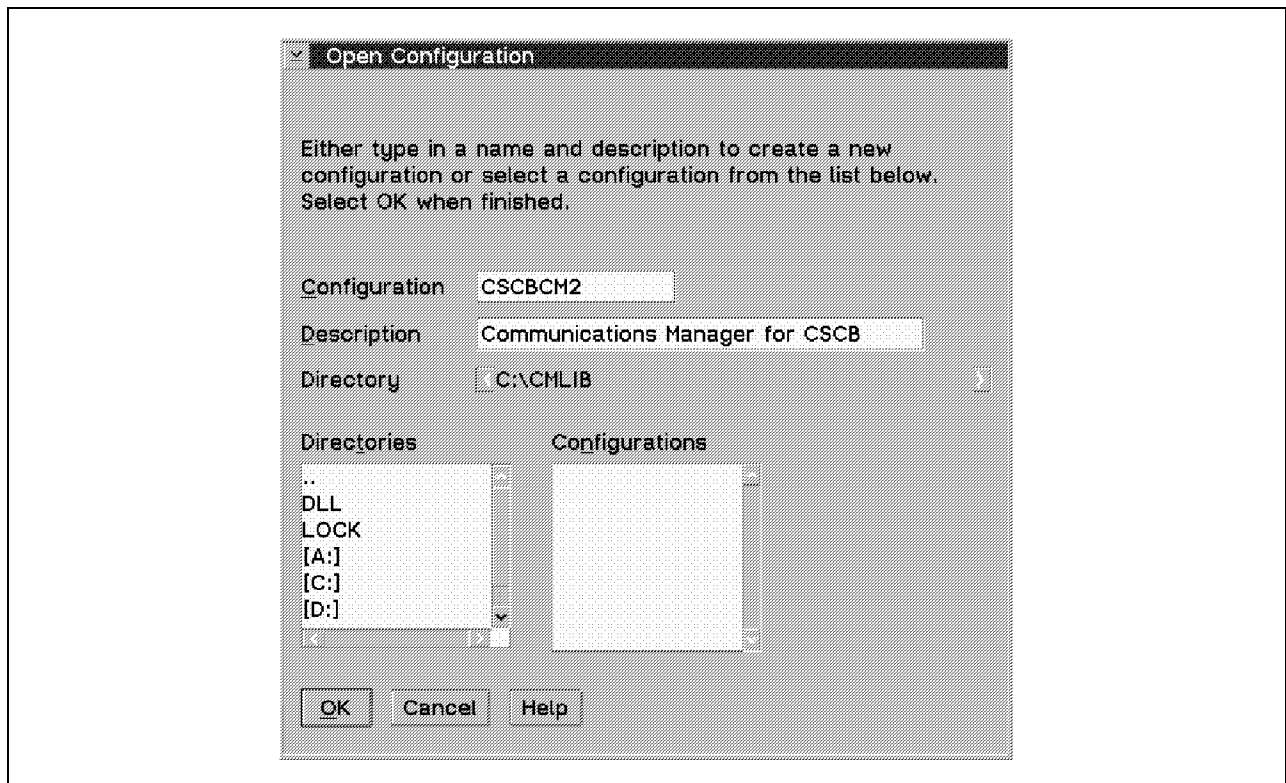


Figure 221. Open Configurations Window

8. On the OS/2 Communications Manager window, click on **Yes** to create the new configuration.
9. On the next OS/2 Communications Manager window (Figure 222 on page 162), click on **Yes** to use the configuration for this workstation.

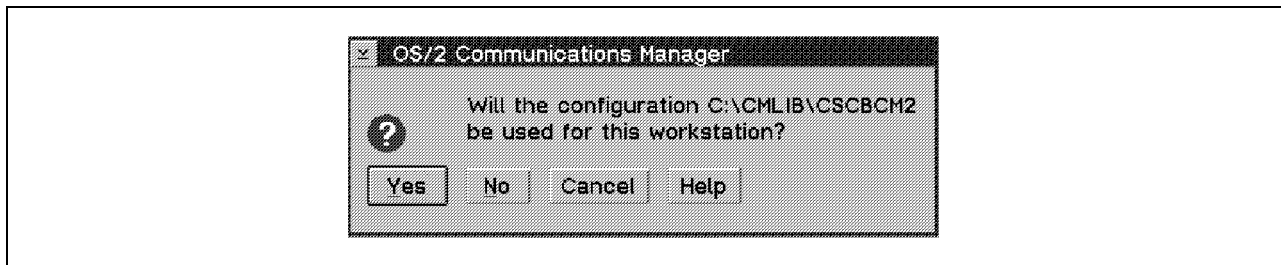


Figure 222. OS/2 Communications Manager Window—Workstation

10. On the Communications Manager Configuration Definition window (Figure 223) you have to define the connection types you want to use. In the field **Additional definitions** select **Token-ring or other LAN types** as the workstation connection type and **3270 emulation** as the feature for the application. Then click on **Configure....**

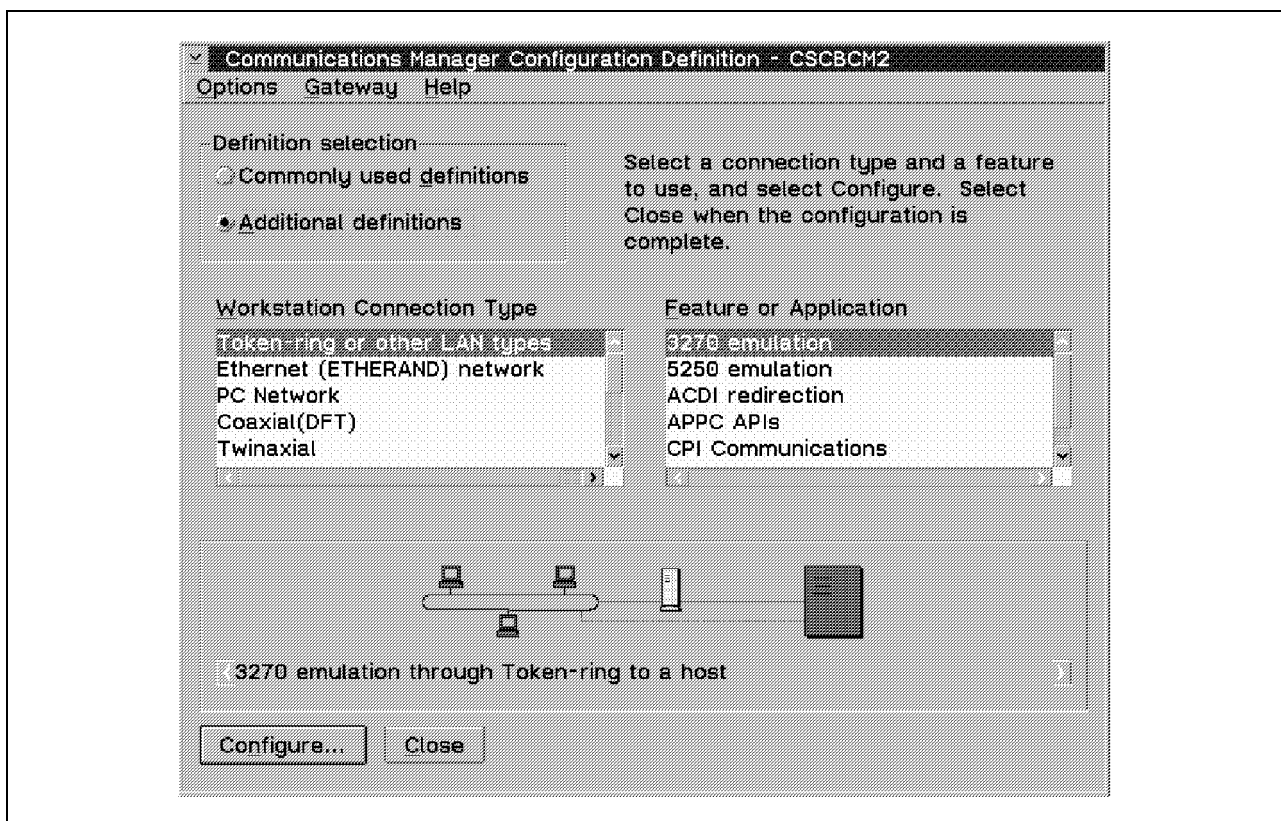


Figure 223. Communications Manager Configuration Definition-CSCBCM2 Window

11. The 3270 Emulation through token-ring window appears (Figure 224 on page 163). You have to specify the following parameter for the emulation sessions:
 - The Network ID is the name of the network where your workstation is located.
 - The Local node name is the name of the workstation as it is known on the network. You are creating a new configuration, so type the name to be assigned to your local node (your workstation). This name becomes the control point name for your node.

- Local node ID is made up of the characters that form the last eight digits used in the exchange identification (XID) for activating a link. The first three characters default to X'05D'. Accept this default. The coordinator of the host computer you connect to can tell you which local node ID to use.
- The LAN destination address is the address of the adapter on your network's communications controller or gateway. For a 3270 emulation configuration, the LAN destination address is the address of the network adapter for your SNA gateway, or your SNA controller.
- The number of terminal sessions can be between one and four.
- Finally, specify the number of printer sessions you want to configure.

Click on **OK**.

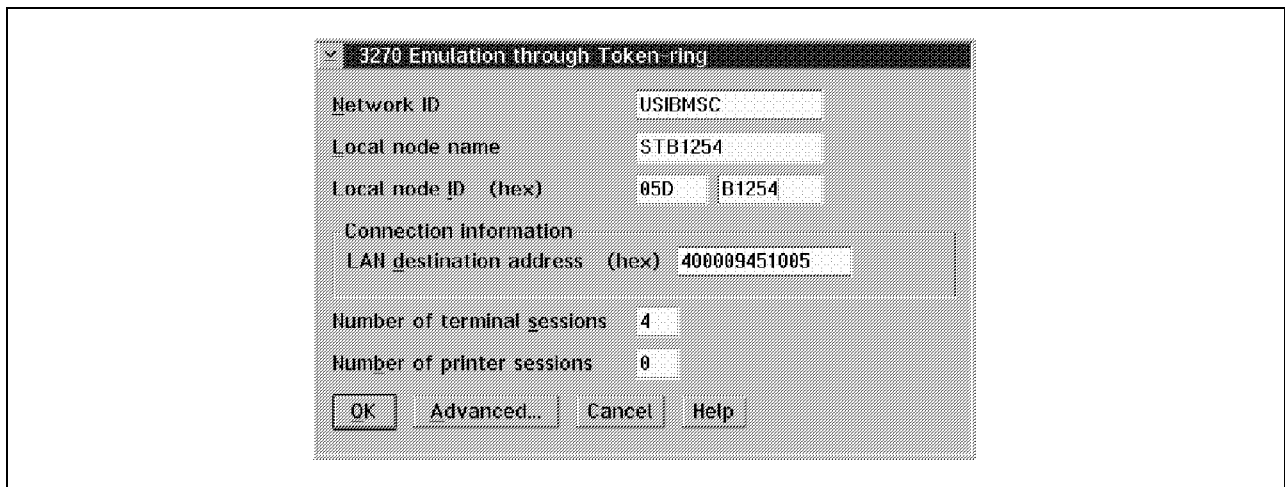


Figure 224. 3270 Emulation through Token-ring Window

12. After the setup of the 3270 emulation, click on **Yes** on the window labeled OS/2 Communications Manager (Figure 225).

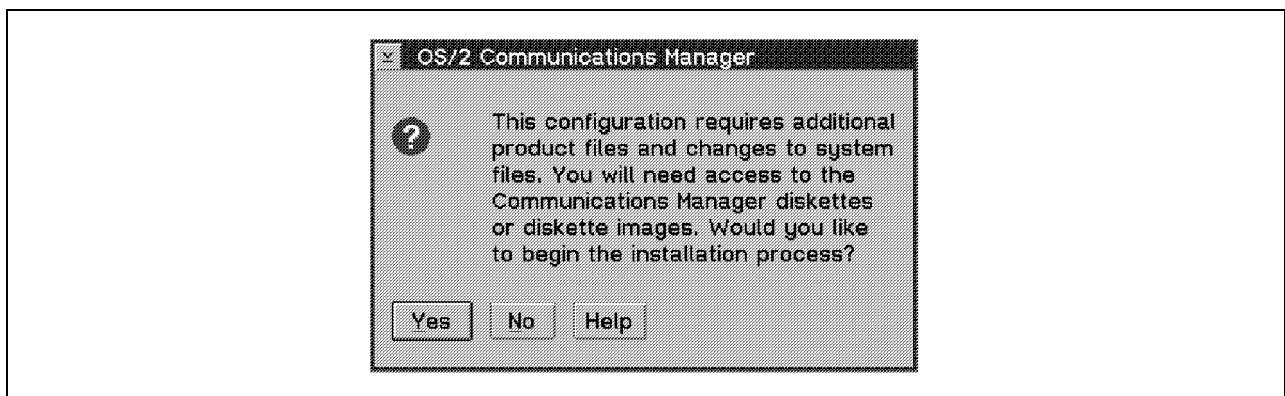


Figure 225. OS/2 Communications Manager Window—Product Files

13. On the Install window, click on **OK** (Figure 226 on page 164).

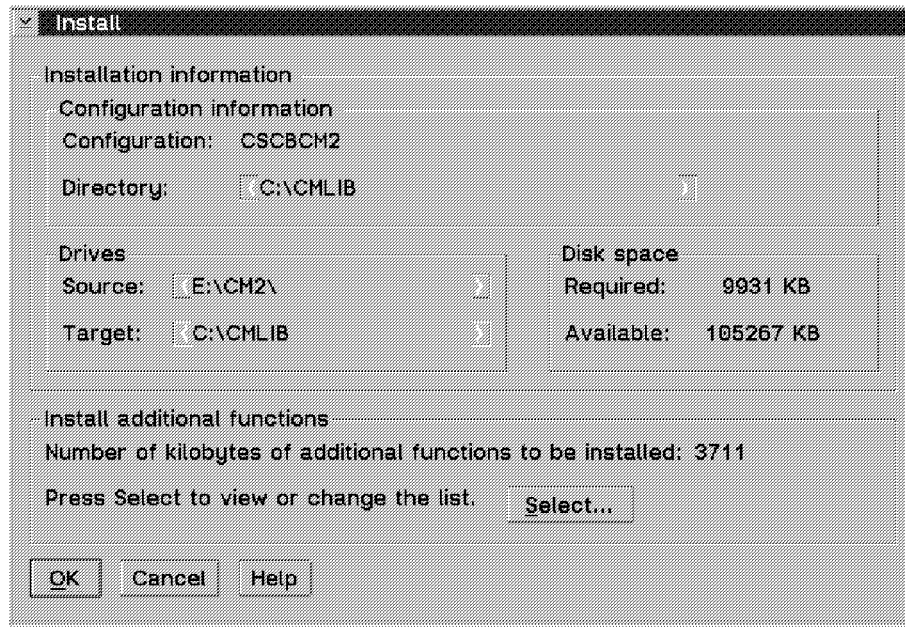


Figure 226. Install Window—Communications Manager/2 for OS/2 Warp

14. On the Change CONFIG.SYS window, accept the default and click on **OK** (Figure 227).

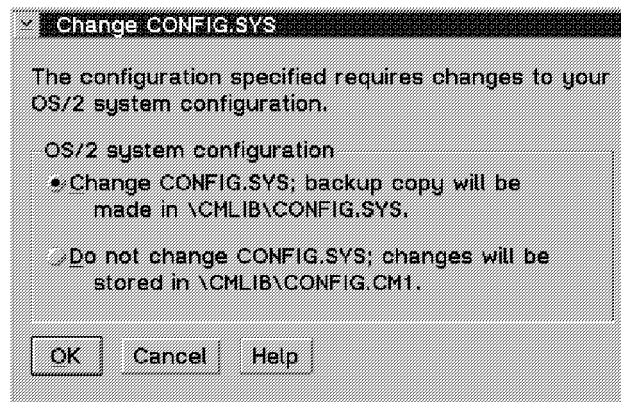


Figure 227. Change CONFIG.SYS Window

15. Close the configuration by clicking on **Close** in the Communications Manager Completion window (Figure 228 on page 165).

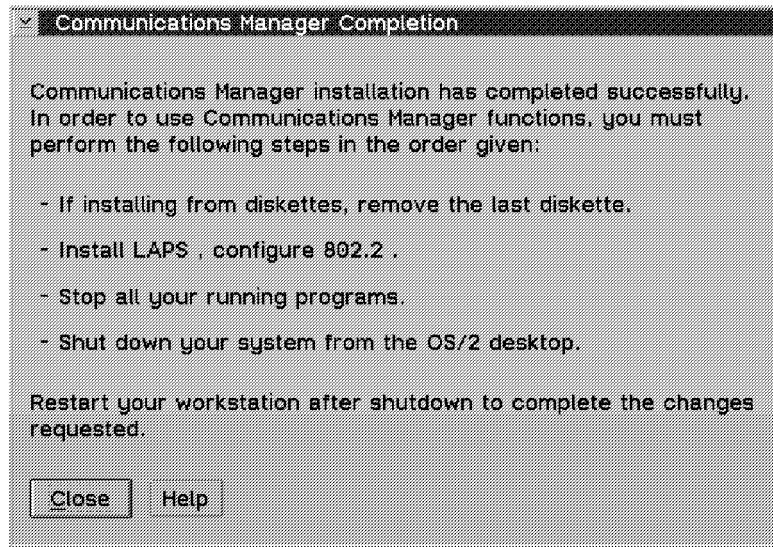


Figure 228. Communications Manager Completion Window

16. You have now finished the installation of Communications Manager/2 for OS/2 Warp. You can find the Communications Manager/2 icon on your desktop. Double-click on this icon to open the Communications Manager/2 - Icon View window (Figure 229).

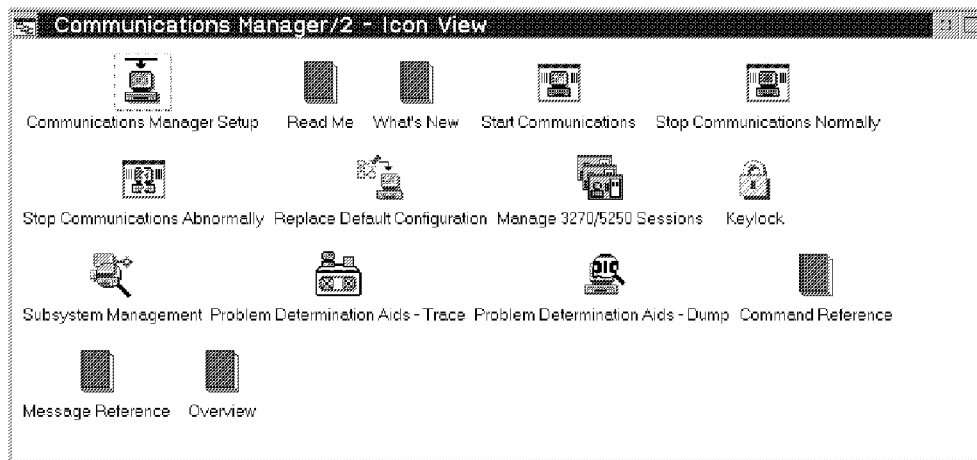


Figure 229. Communications Manager/2 - Icon View Window

17. Shut down your workstation and reboot. To finish the configuration of Communications Manager/2, double-click on the MPTS icon on your Desktop. First click on **OK** on the Multi-Protocol Transport Services-Logo window. Then click on **Configure** on the Multi-Protocol Transport Services window (Figure 230 on page 166).



Figure 230. Multi-Protocol Transport Services Window

18. LAN adapters and protocols may already be configured on your machine, but we need to add a protocol. Ensure, that **LAN adapters and protocols** is selected and click on **Configure** on the Configure window (Figure 231).

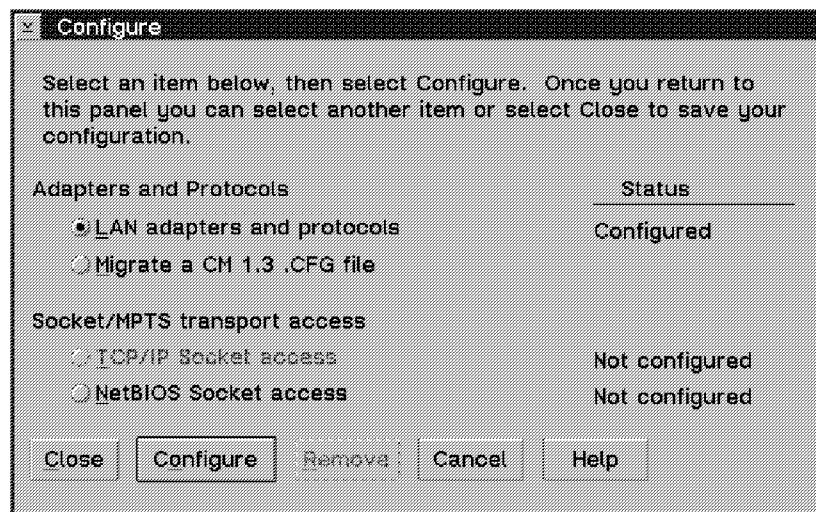


Figure 231. Configure Window—MPTS

19. On the LAPS Configuration window (Figure 232 on page 167), select the network adapter **3270 Adapter for 3174 Peer Communications** and **IBM IEEE 802.2** as its protocol. Click on **Add** in the part of the window for the protocols, and IBM IEEE 802.2 appears in the current configuration list. You need not change the default parameters of this protocol. Click on **OK**

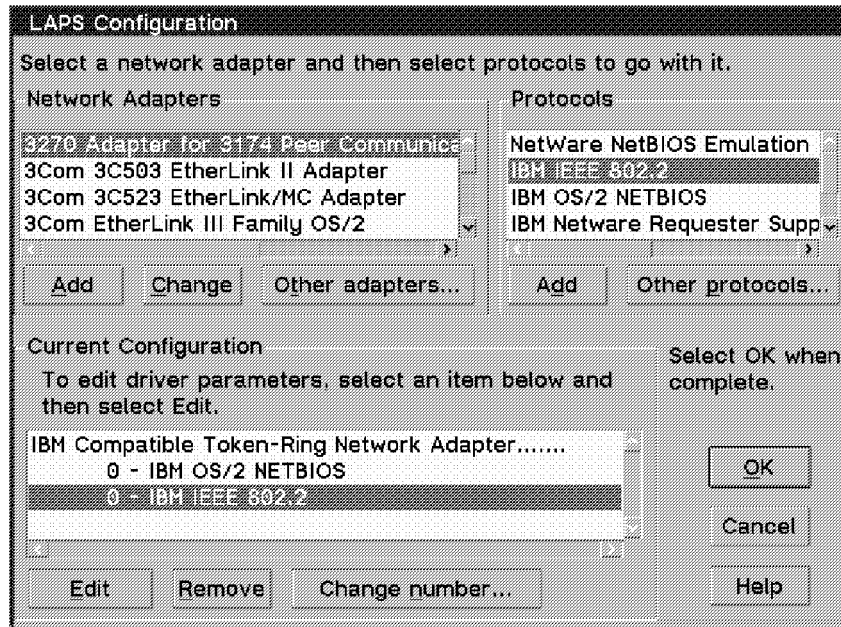


Figure 232. LAPS Configuration Window

20. Figure 231 on page 166 appears again. Click on **Close** and on **Exit** on the next window.
21. Ensure on the Update CONFIG.SYS window (Figure 233) that **update CONFIG.SYS** is selected and click on **Exit**.

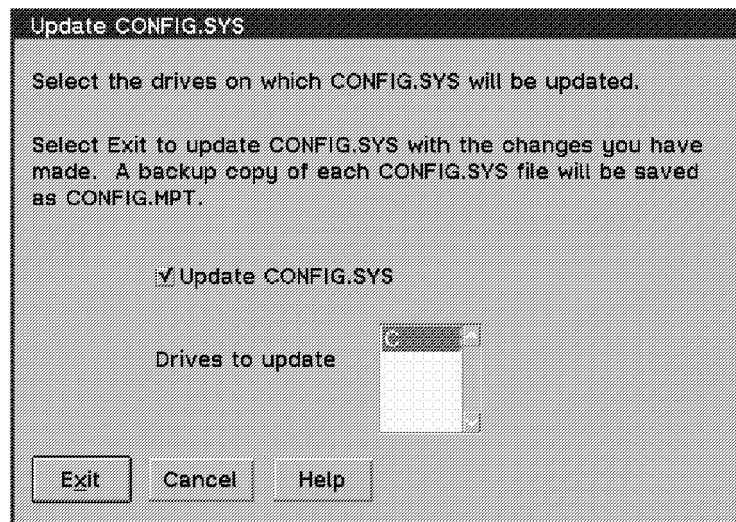


Figure 233. Update CONFIG.SYS Window

22. The message window Update CONFIG.SYS shows you the successful update of the CONFIG.SYS file. Click on **OK** on this window.
23. To exit MPTS click on **Exit** on the Exiting MPTS window.
24. Shut down your workstation.

With the next start of your workstation, the sessions for the 3270 Emulation will appear.

5.2 Download the Necessary Data to the Workstation

If you want to maintain an existing MVS COBOL application on your workstation, you must transfer the program source, including copy books, CICS BMS definitions, and maybe DB2 test data to the workstation.

Because you don't have our sample application on your MVS system, you cannot download its source code members. Instead, copy them from the diskette.

Open an OS/2 window and create the target directories using the following commands (where D: is the drive where IBM VisualAge for COBOL for OS/2 is installed):

```
[D:\] cd IBMCOBOL
[D:\IBMCOBOL] md EMPLLUH
[D:\IBMCOBOL] cd EMPLLUH
[D:\IBMCOBOL\EMPLLUH] md SERVER
[D:\IBMCOBOL\EMPLLUH] md CL3270
```

Put the diskette into the A: drive and type the following commands:

```
COPY A:\EMPLLUH\SERVER D:\IBMCOBOL\EMPLLUH\SERVER
COPY A:\EMPLLUH\CL3270 D:\IBMCOBOL\EMPLLUH\CL3270
```

We assume that you already completed Section 2.3.2, "Running a CICS Client/Server Application with Database Access" on page 84. Otherwise, you also have to run the following commands:

```
COPY A:\EMPLLU D:\IBMCOBOL\EMPLLU
COPY A:\EMPLLU\SERVICE D:\IBMCOBOL\EMPLLU\SERVICE
```

To redevelop or to extend an existing COBOL application on the mainframe, you can use VisualAge for COBOL instead of COBOL for MVS. Only a few changes have to be made to run a host program on the workstation, or to run a workstation program on the host.

To bring the whole application, including the data to access on the workstation, you have to download the application and to export and import the data. Then you can use DB2/2 on the workstation you develop. Your whole development environment is only one single workstation.

On OS/2, you have the benefits of application development on the workstation with useful tools like the transaction assistant, the database assistant, and the debugger. You can develop in a group; you have your own environment and shared resources such as the database only in this user group.

The whole phase of testing the application can be carried out in this environment. At the end of the test phase, upload the finished application back to the mainframe and complete the development by a final test on the host.

5.2.1 Source Code

Once you download the application files to the workstation, you can add new functions to your existing application or you can redevelop it. On the workstation, the application can access DB2 for OS/2 instead of DB2 for MVS.

To download the application, make sure that IBM VisualAge for COBOL for OS/2 and Communications Manager/2 for OS/2 Warp is installed on your workstation. If you want to develop a CICS application on the PC comparable to our previous example EMPLLU, then you need CICS for OS/2 installed on the machine. Follow these instructions to make the installation:

1. Open an emulator session on the workstation on which you want to redevelop the application.
2. Log on to your MVS/TSO host system with your user ID and your password. You then receive the ISPF main menu named ISPF Primary Option Menu.
3. First, create the directories you need on the workstation. Organize the directories so that you have one or more paths for each application, but do not locate more than one application in the same path. Then go back to the ISPF Primary Option Menu.
4. To select the files you want to download, type 3.4 in the command line of the ISPF Primary Option Menu to reach the Data Set List Utility. On this screen, type the name of the dataset where your COBOL source code files are located in the corresponding field **Dsname Level** and press Enter. You get a list of all matching datasets. Here you can read the exact names of the datasets and files you want to download.
5. To use the transfer function given by the Communications Manager, select **Transfer** in the menu bar of this emulation window with your mouse pointer and select **MVS/TSO** in the pull-down menu as the transfer mode.
6. Select **Transfer** in the menu bar again and select **Receive file from host....**
7. The Receive Files from Host (MVS/TSO) window appears. In the **PC Directories**, qualify the path where the application you want to download should reside. You have already created this directory.
8. Now specify the files on the mainframe. To do this, click on the **Edit...** push button on the window labeled Receive files from Host (MVS/TSO) window. In this window labeled Edit Receive List (MVS/TSO), declare the name of each file you want to download.

If the file name on the host has your user ID as the first-level qualifier, then specify the name here without the qualifier, (the default).

If the file name on the host does not have your user ID as the first-level qualifier, then specify the host file name in quotes.

Give the PC file name in the corresponding field and specify **TEXT** as the **Transfer type**. Then click on **Add to list** on this window (Figure 234 on page 170).

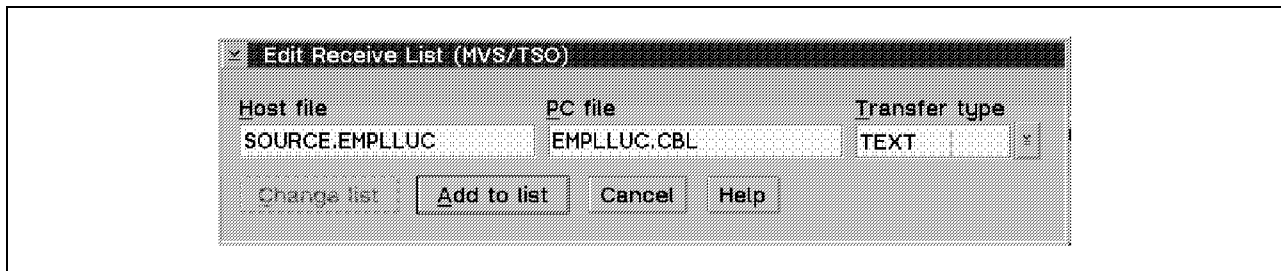


Figure 234. Edit Receive List (MVS/TSO) Window

9. Repeat in this way with each file you want to download. All these files are shown in the list at the bottom of the Receive Files from Host (MVS/TSO) window (Figure 235).

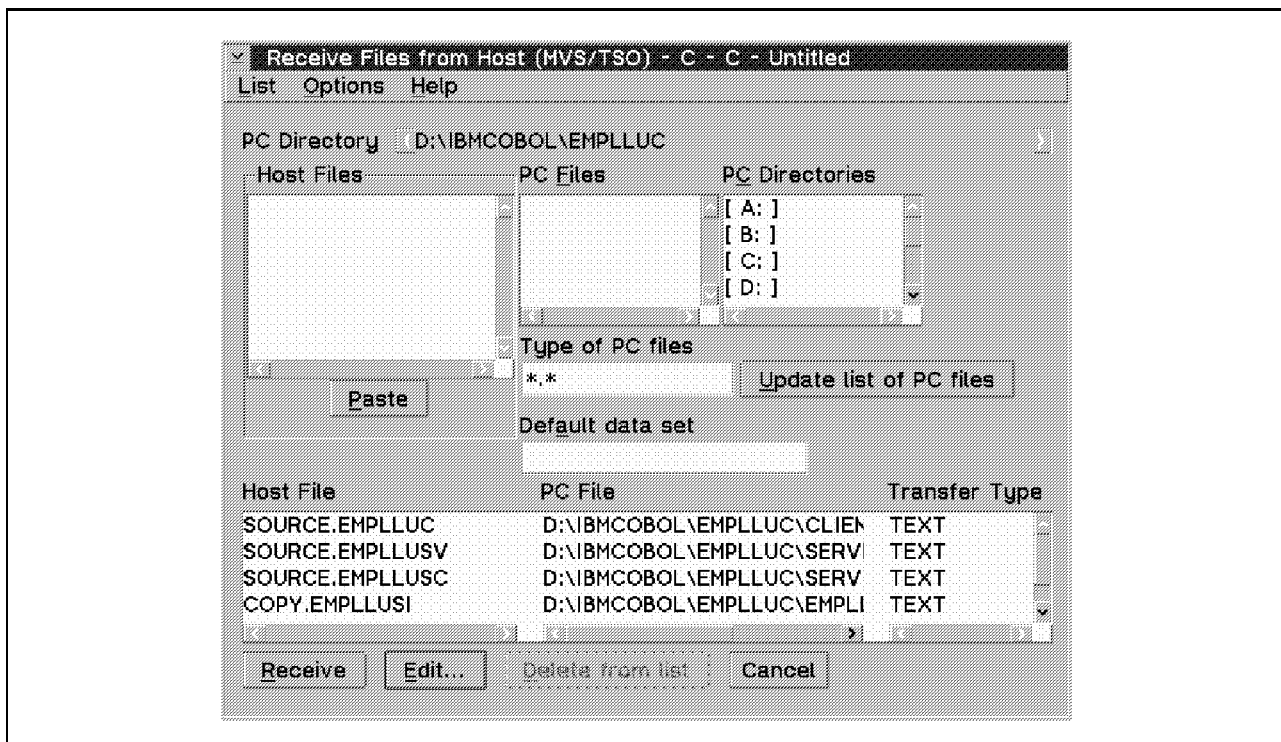


Figure 235. Receive Files from Host (MVS/TSO) Window

10. To run the download, select =6 in the command line of the host emulation and press Enter to get the ISPF Command Shell. This menu can receive data from the workstation.
11. After the transfer has finished, the Receive File(s) Status window disappears. Click on **Cancel** on the Receive Files to Host window to close this window. An OS/2 Communications Manager message window appears and asks you if you want to save the transfer list or not. Click on **Discard** to leave the transfer function without saving the file list. Use the Send Files to Host window to close this window.

The file transfer is finished.

5.2.2 Test Data

If your application accesses DB2 data, you can export your test data on MVS and move it through Distributed Database Connection Services (DDCS) to the workstation. There you can import it to a DB2 for OS/2 database. Or you can use DDCS to access DB2 data on MVS from your workstation. But this way your development environment is not completely on the workstation.

For our sample application, we use a DB2 table that belongs to the sample database provided by DB2, on MVS as well as on OS/2. Therefore we don't have to download any test data nor to create any additional DB2 tables on the workstation. The column definitions of the sample tables are the same on both platforms.

5.3 Prepare the Application on the Workstation

Very few statements related to the DB2 processing are not the same on MVS as on OS/2. Therefore, you have to make some changes in the source code you just downloaded or copied from the diskette in order to compile and run the programs on your workstation.

Sometimes you call common routines from within your COBOL program. If these routines are written in COBOL and you have their source, you can download them to the workstation as well. If not, and this is especially the case for Language Environment routines, you need to use other solutions like replacing the call statement with a constant data assignment. This incompatibility is a reason for making changes in the source code when you up- or download your programs, although the COBOL language itself remains compatible.

5.3.1 CICS 3270 Client Application

Maintaining an MVS CICS online program on the workstation can include the following steps:

- Create the project on the workstation
- Make the necessary changes in the source
- Create the BMS maps for CICS for OS/2
- Define the transaction in CICS for OS/2
- Enhance the program, compile, link, and debug it if needed.

5.3.1.1 Create the Project for the Client on the Workstation

To set up the development environment on your workstation, do the following:

1. Create a COBOL project using the template provided by IBM VisualAge for COBOL for OS/2 as described in the first step of Section 2.3.2.1, "Installing the Server Part of the Application" on page 85.
2. Open the settings of this project. On the **Target** page type `EMPLLUC.DLL` in the *Name* field as shown in Figure 236 on page 172. Select the **Location** tab.

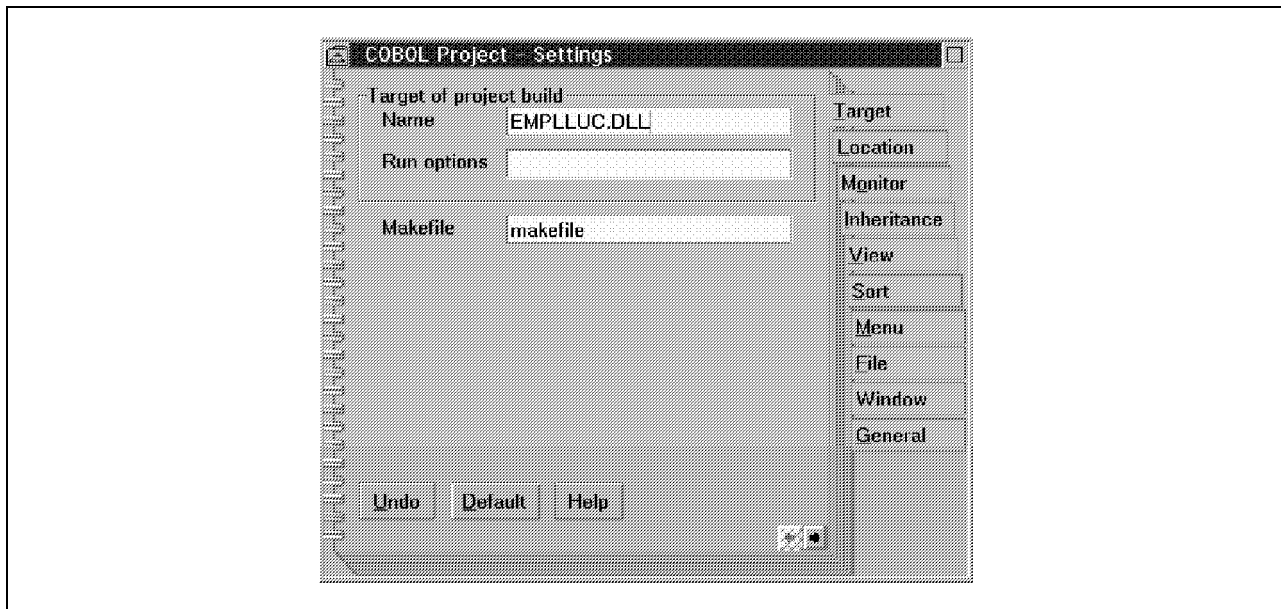


Figure 236. COBOL Project-Settings Window—Target Page (Host Client)

3. The Location page appears (Figure 237). Type `D:\IBMCOBOL\EMPLLU` and `D:\IBMCOBOL\EMPLLUH\CL3270` in the *Source directories for project files* field, where `D:` is the drive where IBM VisualAge for COBOL for OS/2 is installed. Click on the *Working directory* combination box and select `D:\IBMCOBOL\EMPLLUH\CL3270` as the working directory.

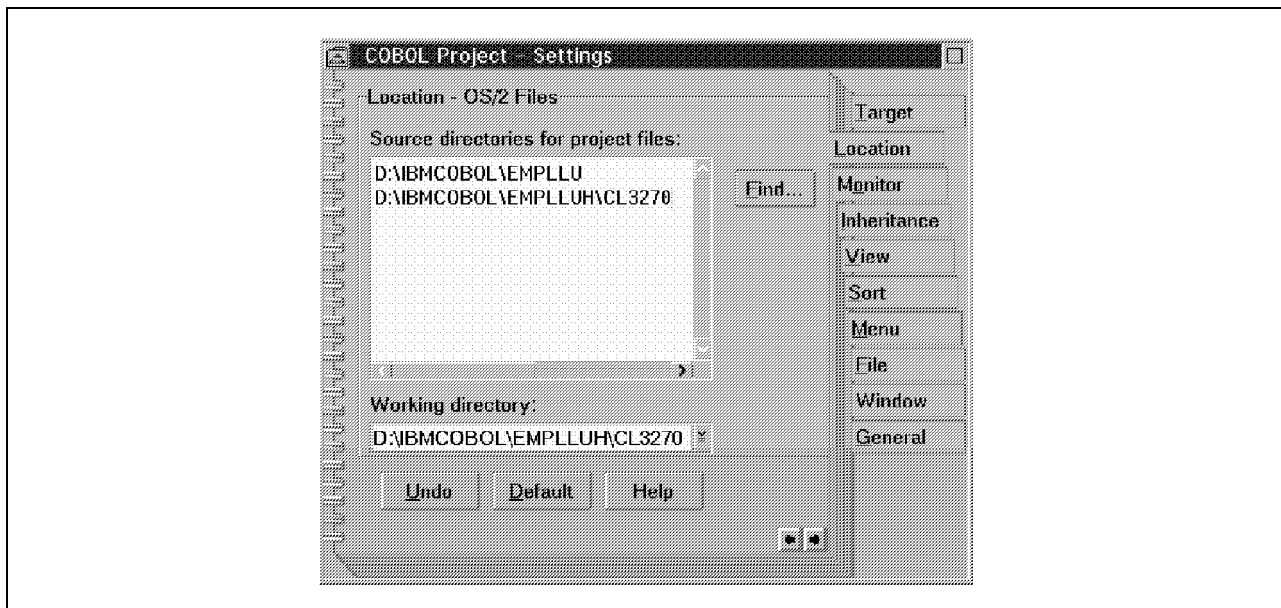


Figure 237. COBOL Project-Settings Window—Location Page (Host Client)

4. Scroll to the **General** page and type `EMPLLU Host Client` in the *Title* field (Figure 238 on page 173). Close the Settings notebook by double-clicking on its system menu symbol.

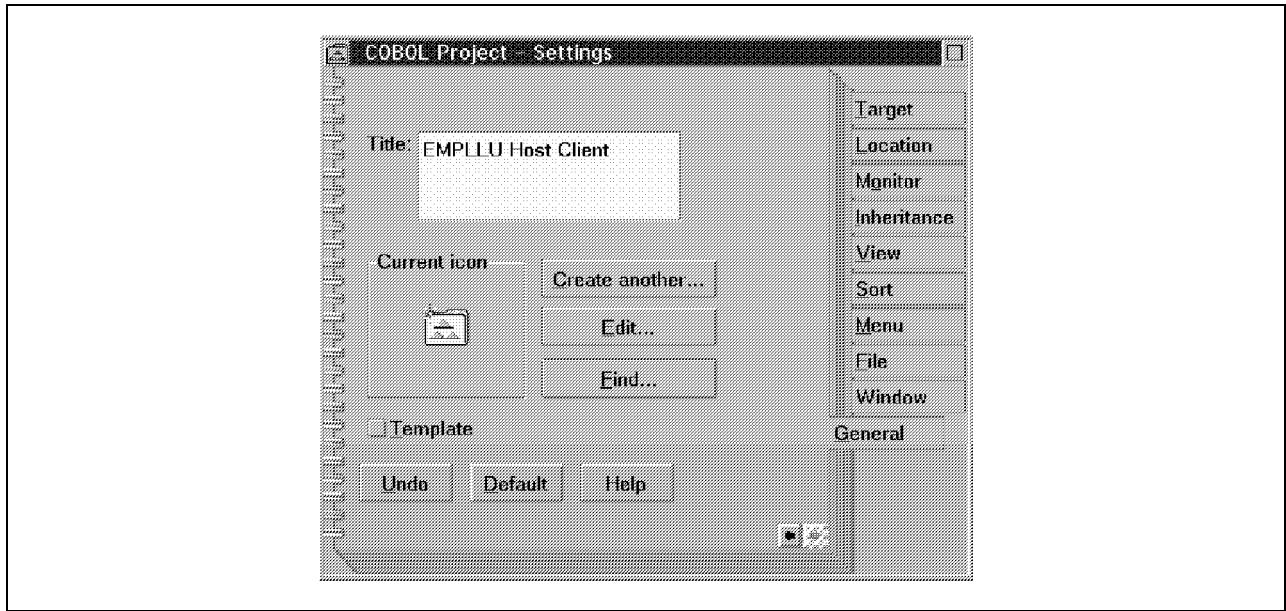


Figure 238. COBOL Project-Settings Window—General Page (Host Client)

5. Open the EMPLLU Host Client project by double-clicking on the **EMPLLU Host Client** icon on the Desktop. The EMPLLU Host Client-Icon view window appears (Figure 239).

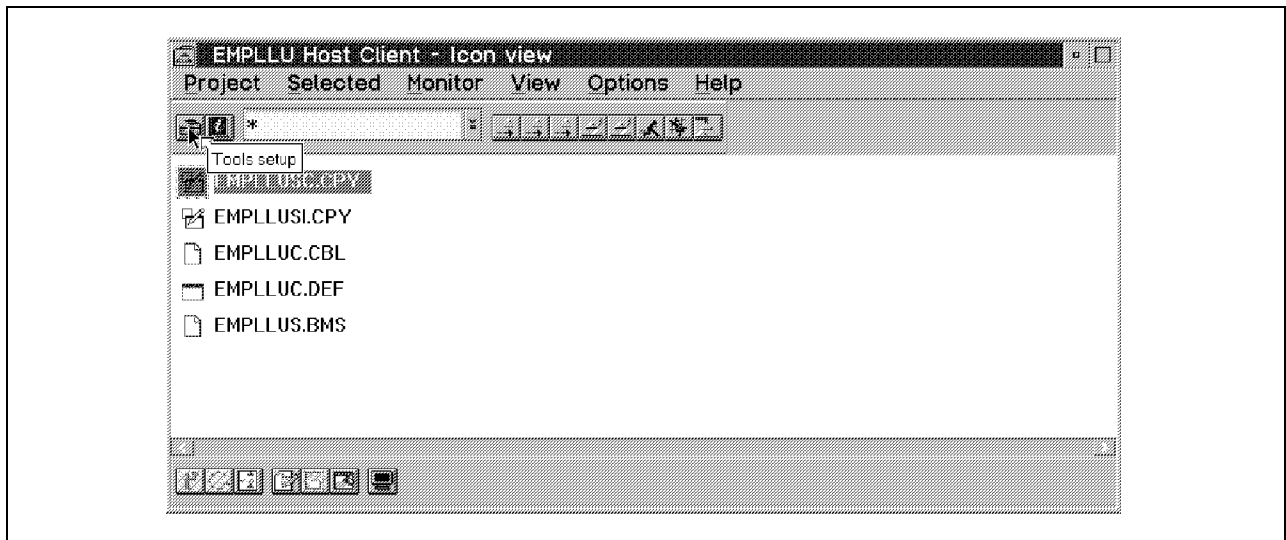


Figure 239. EMPLLU Host Client-Icon View Window

6. Click on the **Tools setup** icon. The EMPLLU Host Client-Tools setup window appears (Figure 240 on page 174).

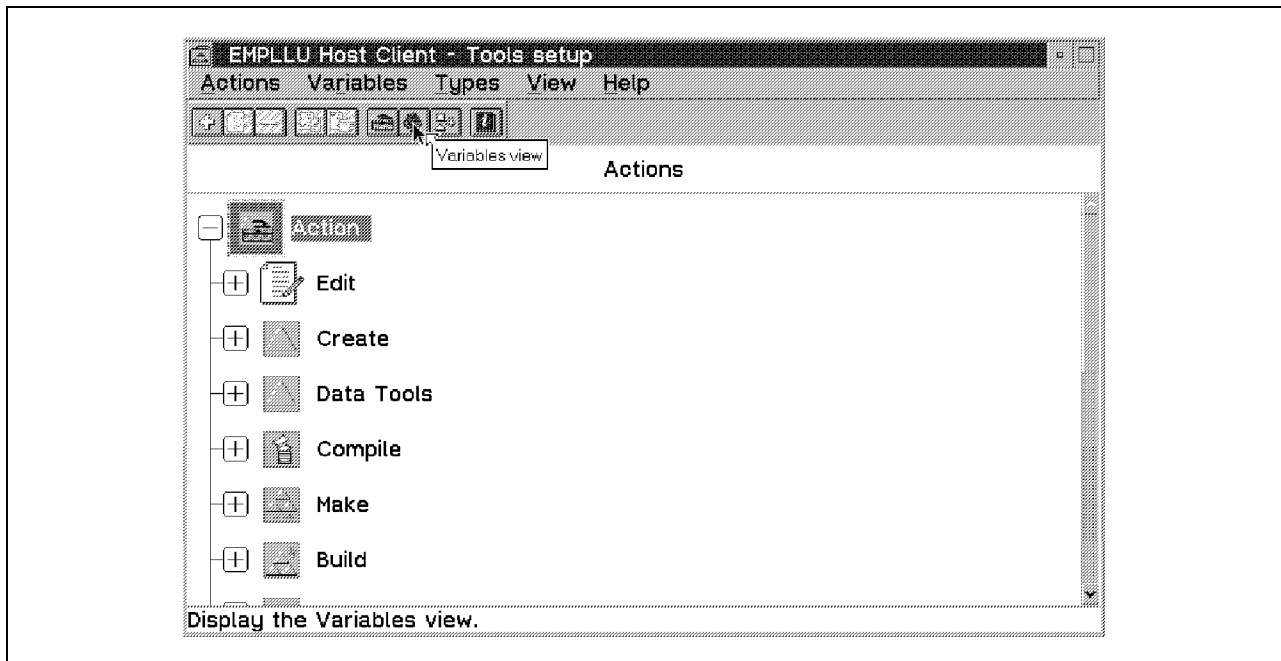


Figure 240. EMPLLU Host Client-Tools Setup Window

7. Click on the **Variables view** icon. The variable view of the EMPLLU Host Client-Tools setup window appears (Figure 241).

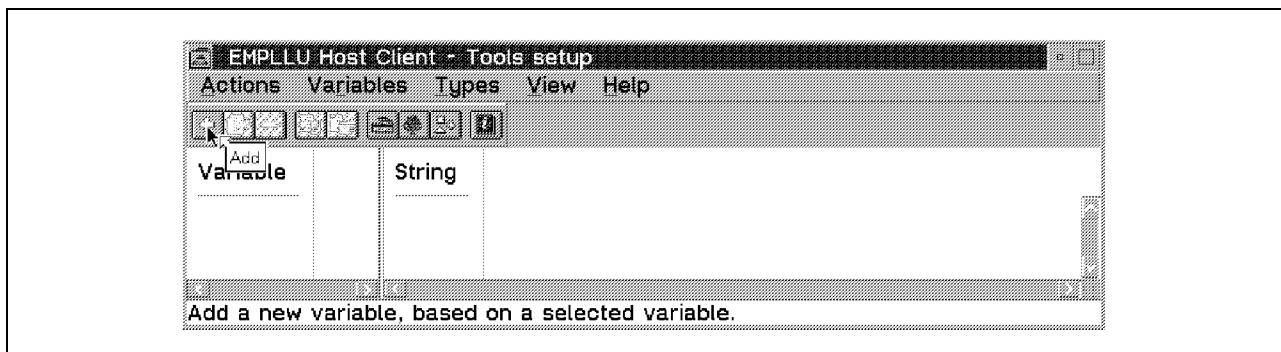


Figure 241. EMPLLU Host Client—Tools Setup Window (Variable View)

8. Click on the **Add** icon.

On the Add Environment Variable window (Figure 242 on page 175), type SYSLIB in the *Name* field and D:\IBMCOBOL\EMPLLU in the *String* field.

\IBMCOBOL\EMPLLU contains the COBOL copy book for the communication area which is being used by the server as well as the client program.

Note: There is one other copy book which is used by the client program: the BMS map input/output structure. This copy book is generated by the CICS map translator, which puts it into the following directory: D:\CICS300\TOOLS\COBOL\COPYBOOK where D: is the drive where CICS for OS/2 is installed. This path is specified in the SYSLIB environment variable of the CICSENV.CMD file. As a result, you don't have to specify the path of the map structure copy book in the project's SYSLIB.

The CICSENV.CMD also sets the LIB environment variable to \CICS300\TOOLS\LIB (where the libraries are that have to be linked

for a CICS program). As a result, you need not specify the LIB variable in the project's tools setup.

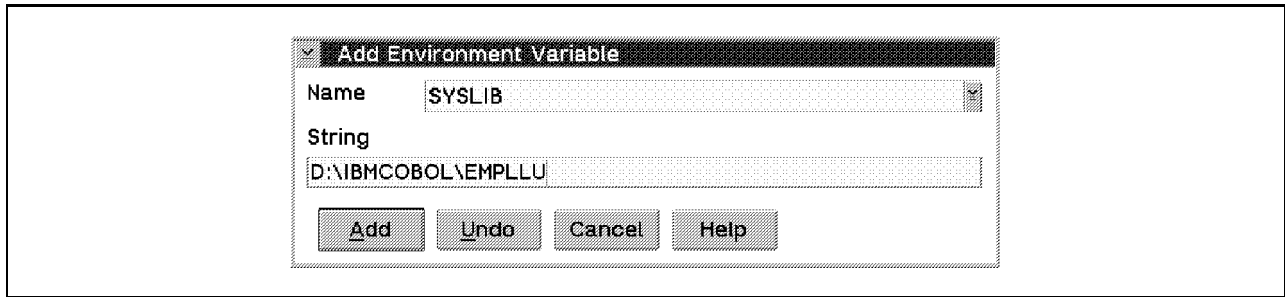


Figure 242. Add Environment Variable Window (Host Client)

9. Click on **Add**. The variable SYSLIB now appears in the variables view of the EMPLLU Host Client-Tools setup.

Close the EMPLLU Host Client-Tools setup window by double-clicking on the system menu icon.

On the EMPLLU Host Client—Icon view window select **Options** from the menu bar and **Compile** from the pull-down menu. Select the **Linktab** of the COBOL Compiler: File scope-IBM COBOL Compiler Options notebook and type `EMPLLUC.DEF` in the *Enter module def file* field, as shown in Figure 243 on page 176.

Note: This module definition file contains a list of all the subprograms in the target DLL of the project that can be called by a program or another DLL. On MVS you don't need such a definition file because there are no DLLs.

If you have an object-oriented class definition program you can use the SOM compiler to generate the module definition file. For a DLL that runs under CICS for OS/2, you have to write the DEF file yourself. Therefore we put it on the diskette even though you cannot download it from the host.

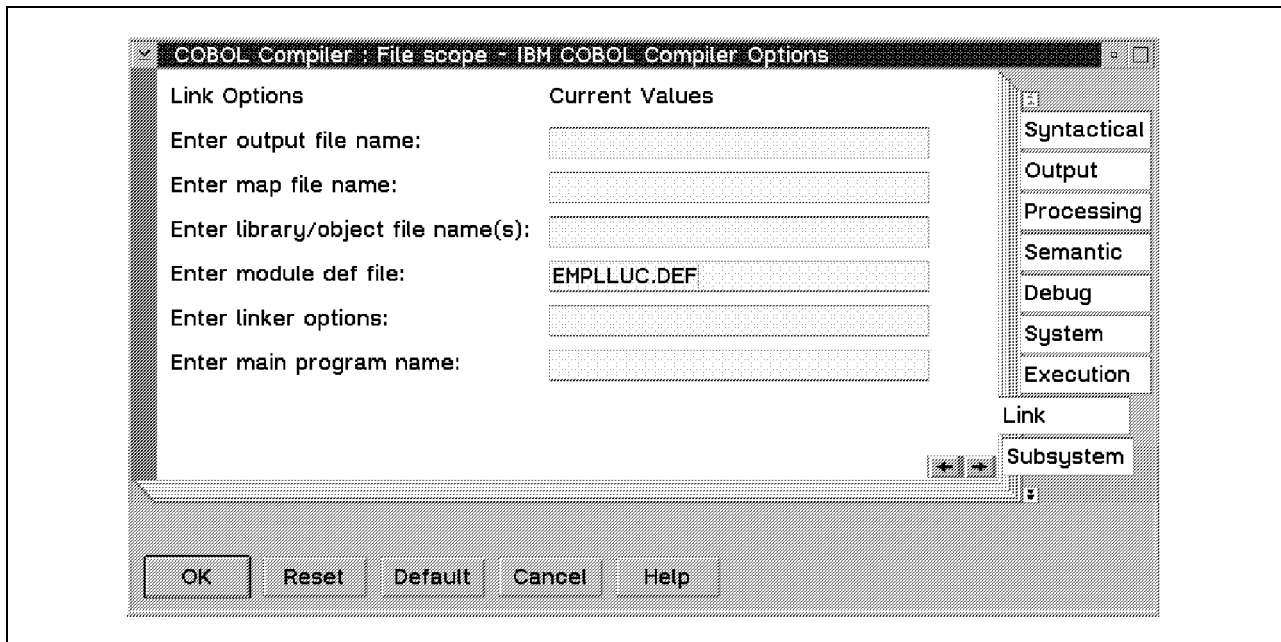


Figure 243. COBOL Compiler: File scope - IBM COBOL Compiler Options Window—Link (Host Client)

10. Scroll to the **Subsystem** page and check the **Preprocess for CICS** check box as shown in Figure 244. Click on **OK** to close the compiler options notebook.

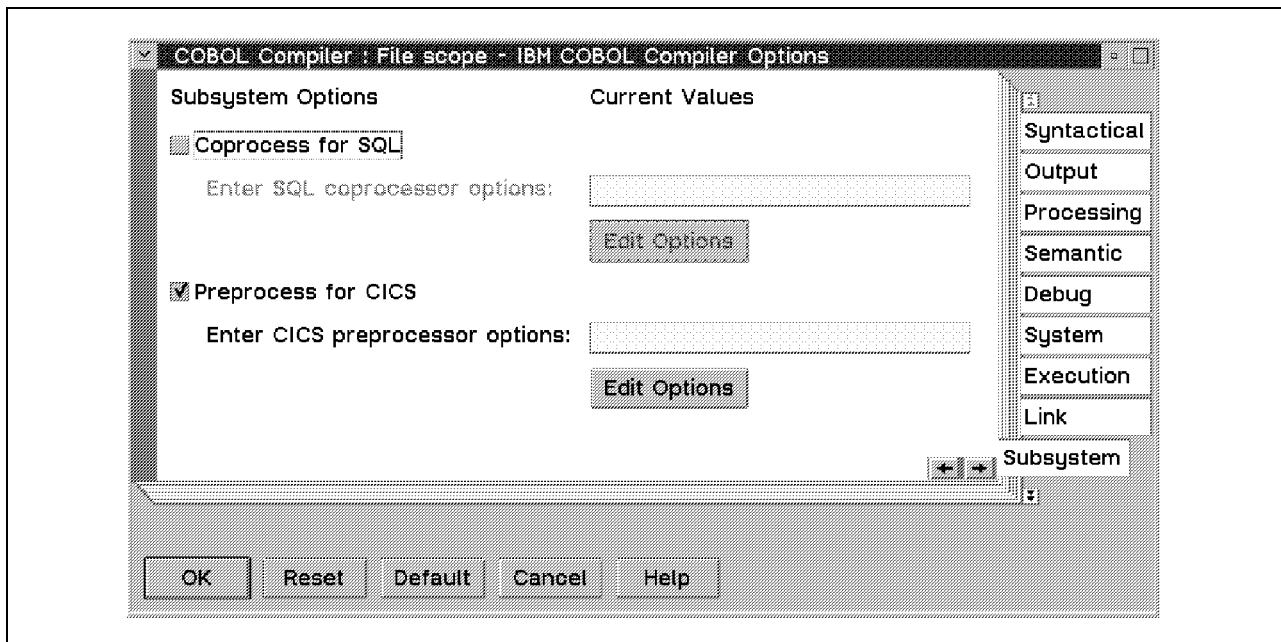


Figure 244. COBOL Compiler: File scope-IBM COBOL Compiler Options Window—Prep (Host Client)

5.3.1.2 Adapt the Source Code of the Client Application

Our sample application is very simple. There are almost no changes to make when you port the application from MVS to OS/2. But be aware that there could be some incompatibilities as our sample shows. You can find more information about this subject in the *IBM VisualAge for COBOL for OS/2 Programming Guide, Chapter 15: Porting Applications between Platforms*.

Besides the fact that you have to create a module definition file on OS/2 in our sample application there is only one change to make in the BMS map definition. The file is called EMPLUS.BMS.

This map set contains two maps: the map to list the employees and a blank map which is displayed when you exit the application. This exit map contains an input field where the user can type the next transaction code. This field is defined with initial blanks. The corresponding statement in the BMS definition is XINIT= the hex code for the blank character. Because the PC uses the ASCII-based character set while the mainframe uses the EBCDIC character set, the value you have to specify for the XINIT parameter is different on both platforms.

Open the EMPLUS.BMS file by double-clicking on the corresponding icon in your EMPLU Host Client-Icon view window. Change the value for the XINIT parameter to 20.

```

:
*   EXIT MAP.
EMPLLEX  DFHMDI SIZE=(24,80),CTRL=(FREEKB),
          COLUMN=1,LINE=1
          DFHMDF POS=(1,1),ATTRB=(UNPROT,NORM,IC),LENGTH=75,
          XINIT=20
:

```

X
X

5.3.1.3 Create the BMS Maps for CICS for OS/2

To translate the BMS map set do the following:

Open an OS/2 window, type:

```
[C:\]cicsmap d:\ibmcobol\emplluh\cl3270\empllus.bms
```

and press the enter key. You will get the messages as shown in Figure 245 on page 178. Note that the command runs the translator twice, once with SYSPARM=DSECT to produce the symbolic description, and once with SYSPARM=MAP to produce the physical map.

```
OS/2 Window
OS/2      Ctrl+Esc = Window List      Type HELP = help
[C:\]cicsmap d:\ibmcobol\emplluh\cl3270\empllus.bms

-----
CICS MAP - Translate D:\IBMCOBOL\EMPLLUH\CL3270\EMPLLUS.BMS
-----

CICS BMS Translator
CICS for OS/2 Version 3.0
(C) Copyright International Business Machines 1988,1995. All Rights Reserved.

Portions (C) Copyright Btrieve Technologies, Inc. 1982,1994. All Rights Reserved.

FAA1989I BMS map translation ended with 0 errors, RC = 0000
CICS BMS Translator
CICS for OS/2 Version 3.0
(C) Copyright International Business Machines 1988,1995. All Rights Reserved.

Portions (C) Copyright Btrieve Technologies, Inc. 1982,1994. All Rights Reserved.

FAA1989I BMS map translation ended with 0 errors, RC = 0000

[C:\]
```

Figure 245. OS/2 Window—CICS MAP Command

The CICS MAP command produces:

- A copy book containing the map input/output structure. This file is called EMPLLUH.CBL and is written to the \CICS300\TOOLS\COBOL\COPYBOOK directory. You find a COPY statement for this copy book in the EMPLLUH.CBL program source.
- The physical map definition that is put into the user map set master file called FAAMSFSC.BTR. This file is held in the \CICS300\RUNTIME\DATA directory and will be accessed by the CICS runtime for the execution of SEND MAP and RECEIVE MAP commands.

You can view how the maps look like by running the CSCA transaction on a CICS for OS/2 terminal.

- A map translator message file called EMPLLUH.TRL. This file is written to the directory where the BMS source is which means that you will see it in your EMPLLU Host Client-Icon view window when you press F5 (refresh). The map translator message file contains statistical information and the corresponding error messages if the CICS MAP command ends with an error.

5.3.1.4 CICS for OS/2 Definitions

The transaction and its initial program have to be defined to CICS for OS/2. If the default definitions for your programs and maps are OK, you don't have to define them in the Processing Program Table (PPT). However, a PPT entry is needed, for example if a program should be on a remote system.

To define the transaction do the following:

1. Start CICS by issuing the CICS RUN command in an OS/2 window or by double-clicking on the **Start CICS D:\CICS300** icon in the CICS for OS/2 Version 3.0 folder.

2. When CICS is started, an IBM logo screen is displayed, as shown in Figure 117 on page 65. Press the Enter key which is the right-hand Ctrl key, as in the host environment. On the next panel, type `sysid` in the *Userid* field and `sysid` in the *Password* field and press Enter.

The message

FAA2250I (CSCBCSV) Signon completed successfully

is displayed. Press Esc to clear the screen, type `CEDA` at the cursor position, and press Enter. The Resource Definition Online panel appears (Figure 246).

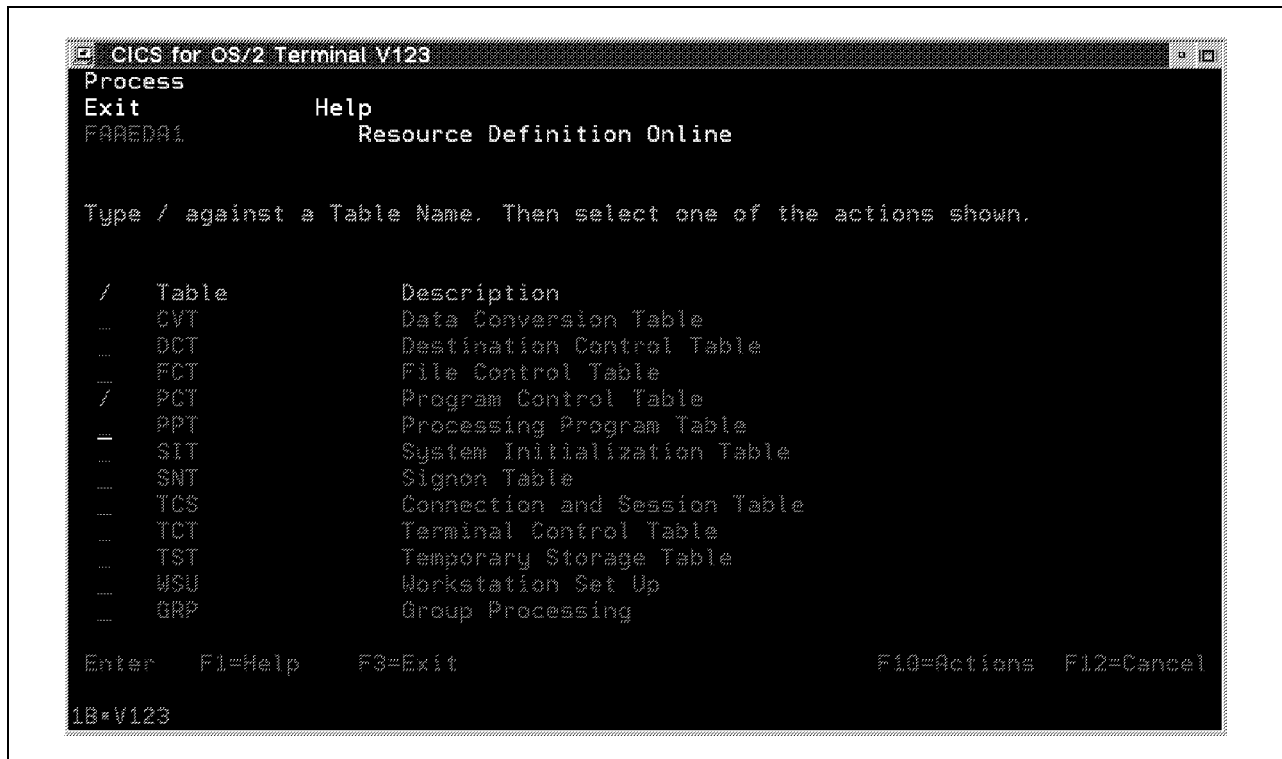


Figure 246. CICS for OS/2 Terminal V123—Resource Definition Online Panel

3. Select the **Program Control Table** by typing a slash in the *PCT* entry field and press Enter. The Program Control Table panel appears (Figure 247 on page 180)

CICS for OS/2 Terminal V123

Update	Add	View	Delete	Search
Exit	Help			

FAAPCT1 Program Control Table More : +

Either type a new Transaction Name and Group Name, or type / against a listed name. Then select one of the actions shown.

/	Transaction	Group	Description
---	AC01	FAASAMPL	ACCOUNTING DEMO AC01
---	AC02	FAASAMPL	ACCOUNTING DEMO AC02
---	AC03	FAASAMPL	ACCOUNTING DEMO AC03
---	AC05	FAASAMPL	ACCOUNTING DEMO AC05
---	ACCT	FAASAMPL	ACCOUNTING DEMO ACCT
---	ACGM	FAASAMPL	SAMPLE LOGO
---	ACLG	FAASAMPL	ACCOUNTING DEMO ACLG
---	ACNU	FAASAMPL	SAMPLE MENU
---	ADDS	FAAIVP	IVP ADD TRANSACTION
---	BRWS	FAAIVP	IVP BROWSE

Transaction Name EMPL Group Name CSCB_____

Enter F1=Help F3=Exit F7=Backward F8=Forward F10=Actions F12=Cancel

1B=V123

Figure 247. CICS for OS/2 Terminal V123—Program Control Table Panel

4. Type EMPL in the *Transaction Name* field and CSCB in the *Group Name* field on the bottom of the panel. Press F10 (Actions) to get the cursor to the action bar. Use the Tab key to move the cursor to the **Add** action and press Enter. The Program Control Table-1 appears (Figure 248).

CICS for OS/2 Terminal V123

Update	Add	View	Delete
Exit	Help		

FAAPCT2 Program Control Table-1

Transaction Code EMPL
 Group Name CSCB_____
 Program Name EMPLLUC__

Secure N (Y or N)
 Can Be Purged. Y (Y or N)
 Dump On Abend. Y (Y or N)
 Priority 0 (0-255)
 Task Class N (1-10 or N)
 Resource Token _____

Use Alternate Screen Size. N (Y or N)

System ID. _____
 Remote Transaction Code. _____

Description. Employee Lookup_____

Enter F1=Help F3=Exit F10=Actions F12=Cancel

1B=V123

Figure 248. CICS for OS/2 Terminal V123—Program Control Table Panel-1

5. Type `EMPLLUC` in the *Program Name* field and optionally a description in the *Description* field. Press Enter. A message is displayed saying that the record has been added successfully. Press F3 twice to exit the Program Control Table.
6. Press Esc to clear the screen and type `CQUIT` to shut down CICS. When you start CICS again, your transaction definition will become active.

5.3.1.5 Build and Run the CICS Client

You can now compile and link the client program. On the EMPLLU Host Client-Icon view window select **Project** from the menu bar and **Build** from the pull-down menu.

Copy the generated EMPLLUC.DLL to a directory specified in the UserWrk variable of the CICSENV.CMD file.

The client program calls the same server program like the GUI client described in Section 2.3.2.2, “Installing the Client Part of the Application” on page 95. Therefore, if you already went through Section 2.3.2, “Running a CICS Client/Server Application with Database Access” on page 84, you don't have to create the server part again before you can run the transaction.

In the next section, however, we describe how to build the server project if you have downloaded the corresponding source files from the host. There are some changes to make.

Because the GUI client program and the 3270 CICS client program provide the same functionality and use exactly the same server program, this is a good example to show what it means if you want to substitute a really event-driven graphical user interface for a character-based user interface. If you compare the two programs, EMPLLUGU.CBL and EMPLLUC.CBL, you see that the presentation client program cannot just be modified a little; it must to be rewritten.

To run the transaction, do the following:

1. Start DB2 by issuing the `DB2START` command in an OS/2 window on the machine where your database is held.

The server program connects to a database with the alias `SAMPLE`. In the DB2 Client Setup on the CICS server machine, you had to define the database with this alias. If you defined it on the local node, the alias points to the sample database of the DB2 for OS/2 - Single-User which you installed on the CICS server machine. In this case, you have to start DB2 on the CICS server machine. If you defined the database on the server node, you have to start DB2 on the DB2 server machine.

If you have followed the instructions in the previous chapters, the database alias is defined on the `CSCBS1` node. Therefore you have to start DB2 on the DB2 server machine.

2. Start CICS by issuing the `CICSRUN` command in an OS/2 window.
3. Press Enter on the IBM logo screen.

When the CICS Sign-on panel is displayed, press Esc twice to get an empty screen. You don't have to log on to CICS because the EMPL transaction has been defined in the Program Control Table with `Secure: N` (No security).

4. Type `empl` as shown in Figure 249 on page 182 and press Enter.



Figure 249. CICS for OS/2 Terminal V123 Window—EMPL transaction

5. The empty Employee Lookup panel appears (Figure 250 on page 183). This panel looks the same as the Employee Lookup GUI window in Section 2.3.2.3, “Running the Application” on page 100, except that there are no push buttons. Instead, in a character-based application, you use the function keys or the Enter key for actions.

Press Enter to display the list.

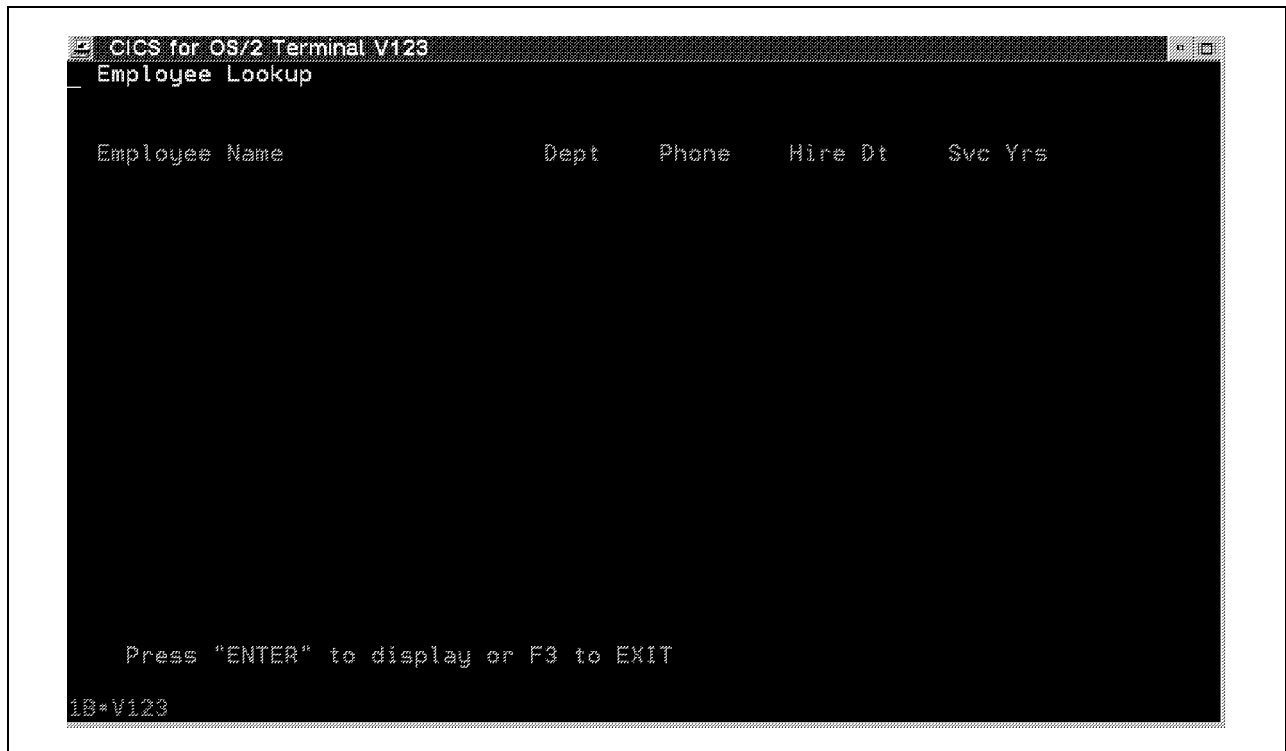


Figure 250. CICS for OS/2 Terminal V123 Window—Employee Lookup Panel

6. If the CICS server workstation is not yet logged on to the DB2 server machine (either to the domain or the NetBIOS node) the Node Logon window appears. In the *User ID* field type the user ID that was used to create the sample database, and in the *Password* field type the corresponding password. In our example, we used CSCBADM as user ID and PASSWORD as password. Click on **OK**.

The Employee Lookup panel is displayed again showing the list of employees (Figure 251 on page 184).

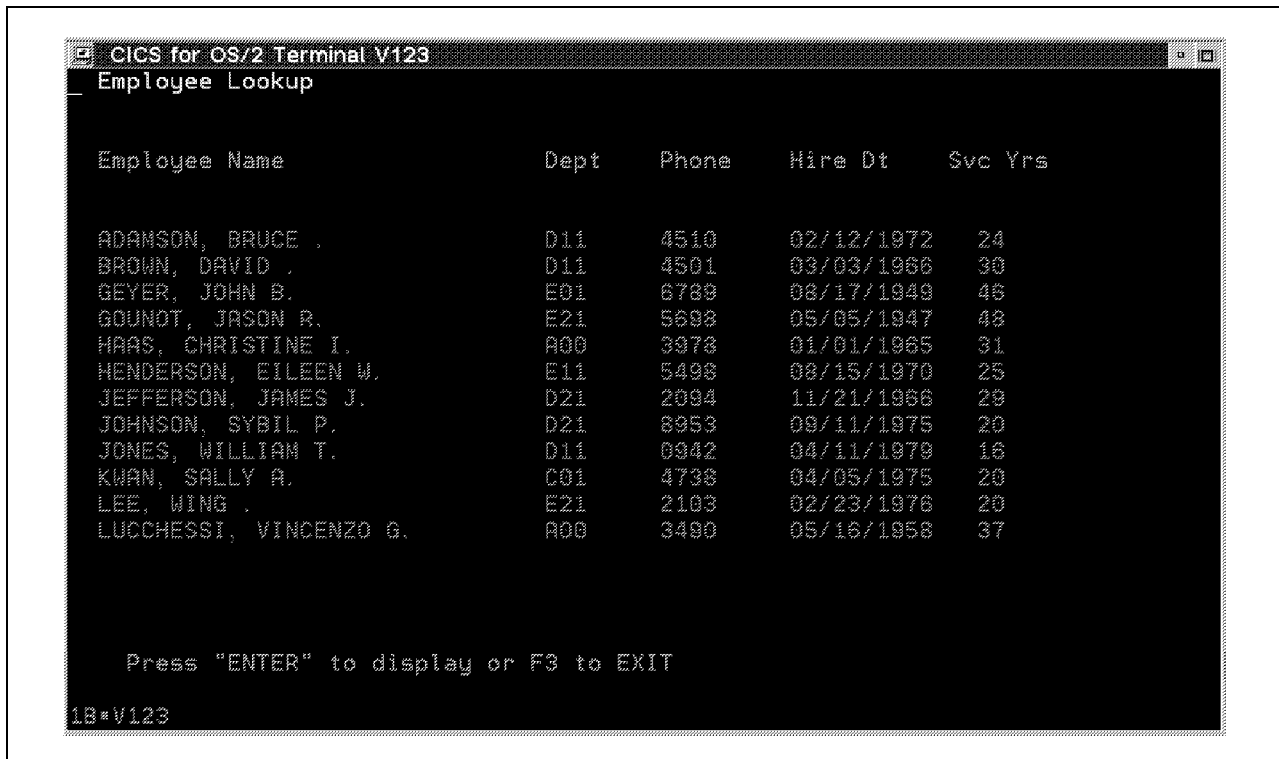


Figure 251. CICS for OS/2 V123 Window—Terminal Employee Lookup Panel, List of Employees on OS/2

- Press F3 to exit. The transaction ends and the empty exit map defined in the EMPLUS map set is displayed. You can enter the next transaction code or CQIT to shut down CICS.

The application is now ported from the MVS environment to OS/2 and running as it is. You can use the OS/2-based setup to change the source code according to the needs, compile, link, and test the modified application using the debugger.

Refer to Section 2.3.4, “Debugging IBM VisualAge for COBOL for OS/2 CICS Programs” on page 106 in order to get information about how to debug a CICS application. For debugging the CICS transaction you have to specify

```
CICSRUN /D-V123 (IDBUG)
```

to start CICS instead of

```
CICSRUN /D-@02@ (IDBUG)
```

5.3.2 CICS-DB2 Server Application

The CICS-DB2 server application is the same as the one used in Section 2.3.2, “Running a CICS Client/Server Application with Database Access” on page 84. However, the source you copied from the diskette, which would be the source you download from your MVS system, is different because the name of the sample DB2 table is different on MVS and on OS/2. Therefore, we explain here what has to be done in order to get an MVS DB2 program to run on an OS/2 workstation. You will create a new project although the target DLL you will build should be exactly the same as the one you built in Chapter 2.

The steps to create the server application include:

- Create the project on the workstation.

- Create the copybook for the DB2 table.
- Make the necessary changes in the source.
- Enhance the program, compile, link, and debug it if needed.

5.3.2.1 Create the Project for the Server on the Workstation

To set up the workframe project for the server go through the steps described in Section 2.3.2.1, “Installing the Server Part of the Application” on page 85 with the following exceptions:

- On the Location page of the COBOL Project—Settings notebook type the following in the *Source directories for project files* field:

```
D:\IBMCOBOL\EMPLLU
D:\IBMCOBOL\EMPLLUH\SERVER
```
- Select D:\IBMCOBOL\EMPLLUH\SERVER in the *Working directory* combination box on the Location page of the COBOL Project—Settings. Because you already created the source directories when you copied the files from the diskette, the Create directories window does not appear.
- Type EMPLLU Host Server in the *Title* field on the General page.
- No changes need be made in the source code of the service calculation routine when you port it from MVS to OS/2. Therefore, you don't have to create another project for the service calculation routine.

Let us assume that the service calculation routine is a standard module called by several programs. In this case, you don't have to define the project because this routine as a subproject in every project that links the module. You would compile the routine once and specify the corresponding object module in the linker option of the calling program, as well as the corresponding directory for the SYSLIB variable in the tools setup.

- You need not copy any files from the diskette because you copied them before (instead of downloading the files).

Add the LIB and SYSLIB variables in the Tools setup and specify the compiler options for the EMPLLU Host Server project.

A date as it is retrieved from DB2 can have different formats. On MVS, this format is defined through an installation parameter. On OS/2 the format depends on the COUNTRY definition in the CONFIG.SYS file. Therefore, the date format on your MVS system could be different from the format on your OS/2 system.

With the corresponding DB2 precompiler option, you can specify what date format you want your application program to get from DB2.

Because the Service Calculation routine of our sample EMPLLU application is calculating with the date it retrieves from DB2, it is important that it get this date in the correct format. If you downloaded the application from the host, you might have had to specify the DATETIME SQL co-processor option for the server project with the parameter of your host date format. But because the server and service calculation routines have been used in previous chapters, you don't have to specify anything special here at this time.

5.3.2.2 Create the Copybook for the DB2 Table

The copybook containing the structure of the DB2 table can be generated on MVS through the DCLGEN utility of the DB2 Interactive services. This copybook contains statements that are not used with DB2 for OS/2. Therefore, you must create a new copybook on OS/2.

You can use the Data Assistant tool to do this:

1. Open the VisualAge for COBOL folder by double-clicking on the VisualAge for COBOL icon on the desktop.
2. Open the Works folder by double-clicking on the Works icon in the VisualAge for COBOL-Icon View window.
3. Double-click on the Data Assistant icon to open the Data Assistant-Icon View.
4. Double-click on the Database Schema view icon. The Database Schema view window appears (Figure 252). Type *SAMPLE* in the *Enter database name* field and click on **OK**.

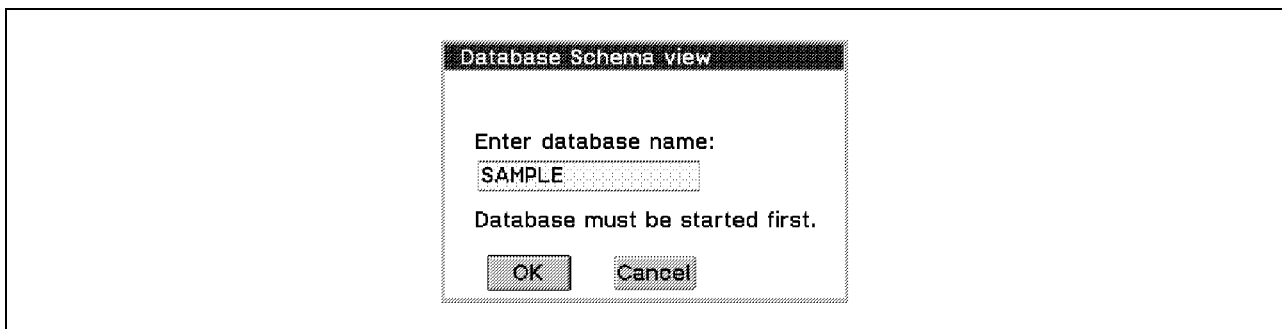


Figure 252. Database Schema view Window

5. If you are not yet logged on to the DB2 server machine (either to the domain or the NetBIOS node) the Node Logon window appears. In the *User ID* field type the user ID which has been used to create the sample database, and in the *Password* field type the corresponding password. In our example, we used CSCBADM as user ID and PASSWORD as password. Click on **OK**.
6. The SAMPLE-Schema window appears (Figure 253).

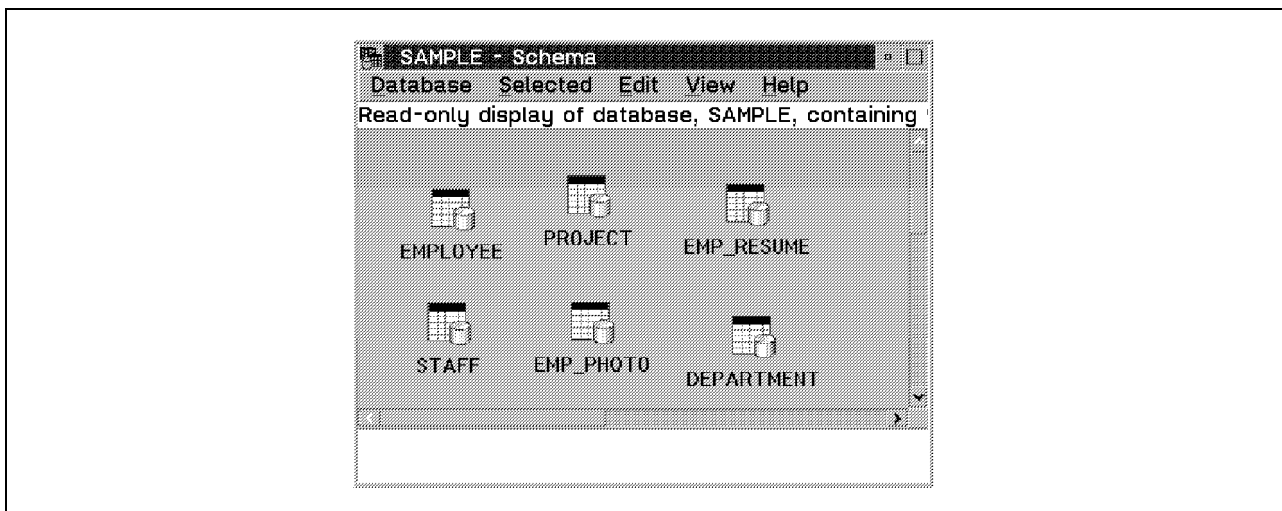


Figure 253. SAMPLE-Schema Window

7. Double-click on the Data Structure Mapping view icon in the Data Assistant-Icon View window. The Data Structure Mapping view window appears (Figure 254). Type **EMPLLU** in the *Enter mapping filename* field and click on **OK**.

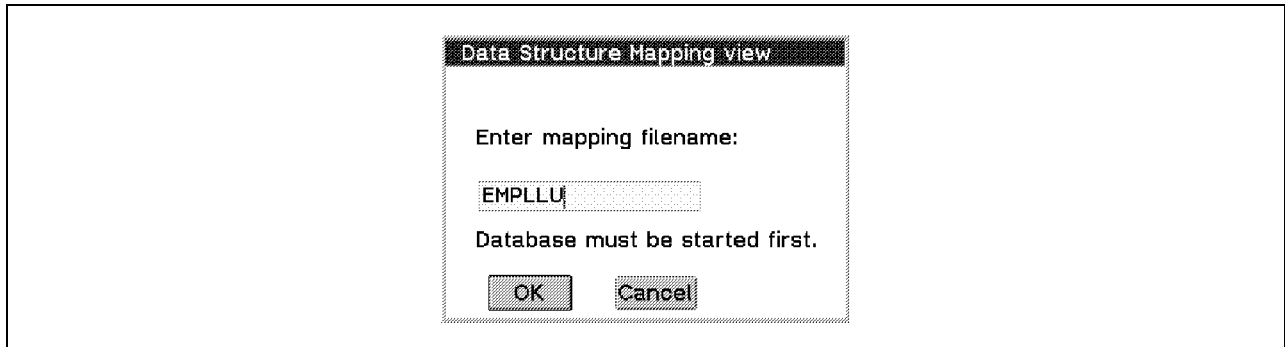


Figure 254. Data Structure Mapping view Window

8. The EMPLLU-Mapping window appears (Figure 255).

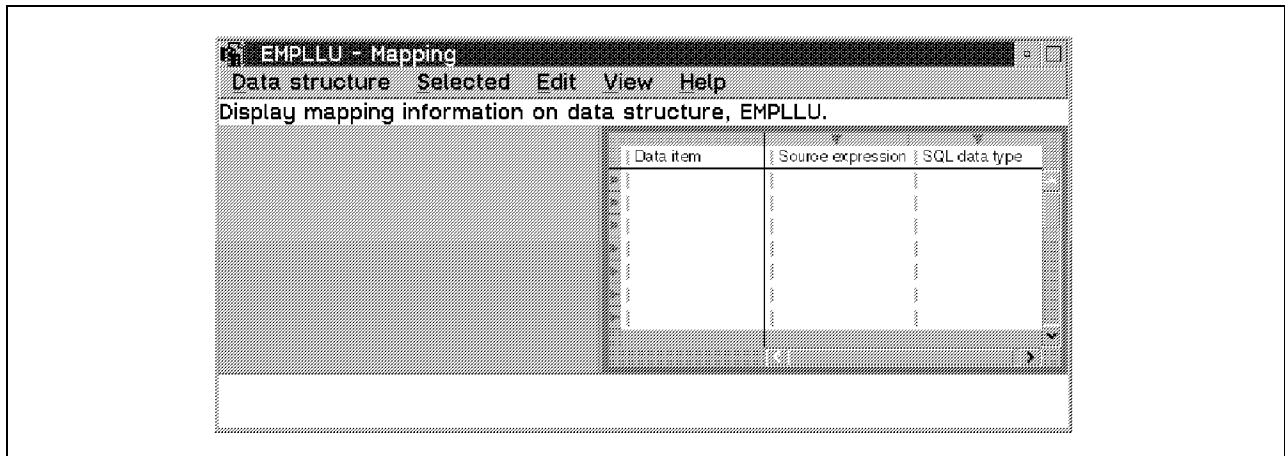


Figure 255. EMPLLU-Mapping Window

9. Move the mouse pointer over the **EMPLOYEE** database icon in the SAMPLE-Schema window and press and hold down mouse button 2. Drag the object from the SAMPLE-Schema window over to the EMPLLU-Mapping window and release mouse button 2.
10. Double-click on the **EMPLOYEE** database icon in the EMPLLU-Mapping window to show the column names of the table.
11. Click on the following column names to select them for the mapping:
 - FIRSTNAME
 - MIDINIT
 - LASTNAME
 - WORKDEPT
 - PHONENO
 - HIREDATE

The Data item table in the EMPLLU-Mapping window is populated as shown in Figure 256 on page 188.

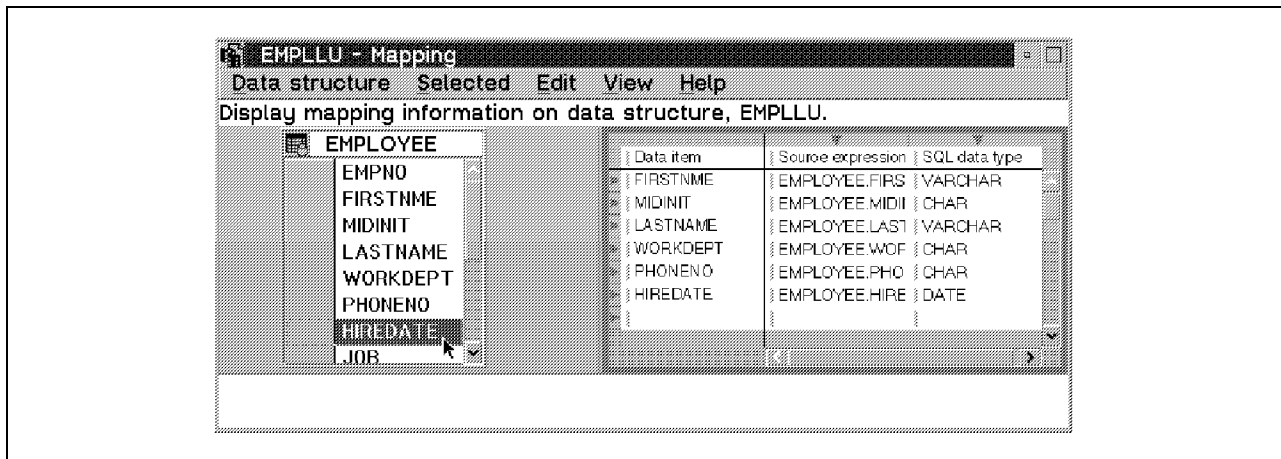


Figure 256. EMPLLU-Mapping Window—Expanded View

12. On the EMPLLU-Mapping window select **Data structure** from the menu bar and **Save** from the pull-down menu. An information message tells you that the copybook has been generated (Figure 257). Click on **OK**.

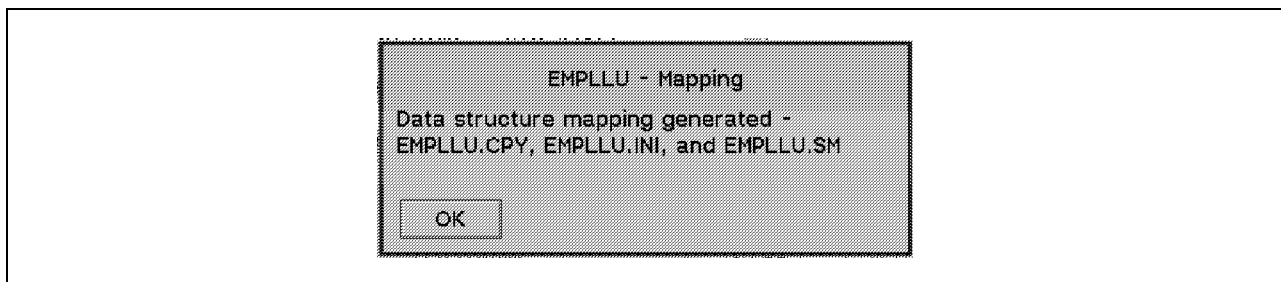


Figure 257. EMPLLU-Mapping Message Window

13. Close the EMPLLU-Mapping window and the SAMPLE-Schema window by double-clicking on the corresponding system menu icons. When you exit from the EMPLLU-Mapping window, you are asked if you want to save before exiting. Click on **Exit**.

The Data Assistant writes the generated files to the `D:\IBMCOBOL\DA` directory. Copy the copybook from there to your project's directory by issuing the following command in an OS/2 window:

```
copy D:\IBMCOBOL\DA\EMPLLU.CPY D:\IBMCOBOL\EMPLLUH\SERVER
```

5.3.2.3 Adapt the Source Code of the Server Application

You need to change some statements in the COBOL program source.

Double-click on the **EMPLLUSV.CBL** icon in your EMPLLU Host Server-Icon view window to start the editor. Make the following changes in the source code to adapt the program to the DB2 for OS/2 coprocessor:

- Change the `FROM` clause in the `DECLARE CSR1 CURSOR` statement to:

```
FROM EMPLOYEE
```

This is because the table names of the sample database are not all the same on MVS as on OS/2.

- At the beginning of the `1000-MAIN` section, add the following statement:

```
EXEC SQL CONNECT TO SAMPLE END-EXEC.
```

- Change the first statement of the 1130-FETCH-ALL section to:

```
INITIALIZE EMPLLU.
```

This is the variable you just generated through the Data Assistant.

Save these changes and close the source file.

5.3.2.4 Build the Server Application

You can now compile and link the server program. On the EMPLLU Host Server-Icon view window select **Project** from the menu bar and **Build** from the pull-down menu.

Copy the generated EMPLLUSV.DLL to a directory specified in the UserWrk variable of the CICSENV.CMD file. When you run the EMPL transaction on a CICS terminal, the CICS client program will call the server program you just built.

The server application is now ported from the MVS environment to OS/2 and running as it is. You can use the OS/2-based setup to change the source code according to the needs, compile, link, and test the modified application using the debugger.

5.4 Upload the Necessary Data to the MVS System

If you upload the application files to the host, then the application can run on the mainframe accessing DB2 for MVS instead of DB2 for OS/2. You can share the program very easily with many other users and you need not install it on each workstation.

To upload the application, do the following:

1. Open an emulator session on the workstation where you have developed the application.
2. Log on to your MVS/TSO host system with your user ID and your password. You receive the ISPF main menu named ISPF Primary Option Menu.
3. First, ensure that the datasets are allocated to where you want to copy the files. Then go back to the ISPF Primary Option Menu.
4. To upload the files from the workstation to the host, type **6** in the command line and press Enter to get the ISPF Command Shell. This menu can receive data from the workstation.
5. Select **Transfer** in the menu bar of this emulation window with your mouse pointer and select **MVS/TSO** in the pull-down menu as the transfer mode. Depending on this mode, the host file name created by the function that sends the files differs.
6. Select **Transfer** in the menu bar again and select **Send file to host...**
7. The Send Files to Host (MVS/TSO) window appears. In the **PC Directories** you have to specify the path where your application you want to upload resides. If you follow our example, you want to load the EMPLLU application to the host. To do so, insert the diskette into drive A and select the Send Files to Host window [**A:**] in the **PC Directories** list box, with a double-click. **A:** appears in the PC Directory field. Then scroll down in the list box of **PC Directories** until you reach the directory [**EMPLLU**]. Double-click on it. Now

this path is added to the PC Directory field and in the **PC Files** list box, the files included in this directory are shown.

Click on **EMPLLUSC.CPY** and **EMPLLUSI.CPY**. Both files are shown now in the list box in the bottom of this window (Figure 258).

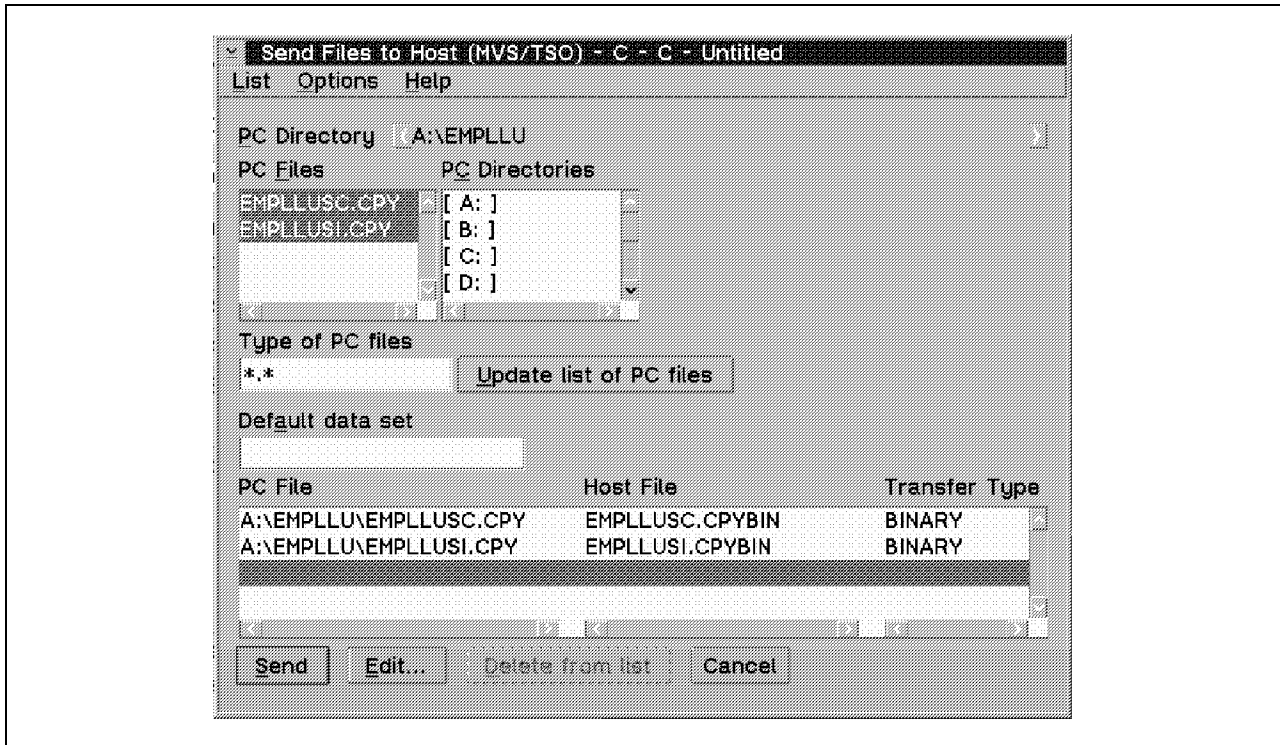


Figure 258. Send Files to Host (MVS/TSO) Window—C-C-Untitled

8. Scroll down the **PC Directories** list box until you reach the directory [CL3270]. Select it by double-click and you can see three files in the **PC Files** list box. Click on **EMPLLUC.CBL** and **EMPLLUS.BMS** to add them into the list for sending to the host.
9. To leave the actual subdirectory on the A drive scroll down in the list box for the **PC Directories** and double-click on [..] to reach the directory A:\EMPLLU again. Scroll down in the same list box to [SERVER] and double-click on it. Add the files **EMPLLU.CPY** and **EMPLLUSV.CBL** to the list for the file transfer in the usual way, by selecting them.
10. Continue in the described way to reach the path **A:\EMPLLU\SERVICE** and select then the file **EMPLLUSC.CBL** in the **PC Files** list box to add it to the transfer files list (Figure 259 on page 191).

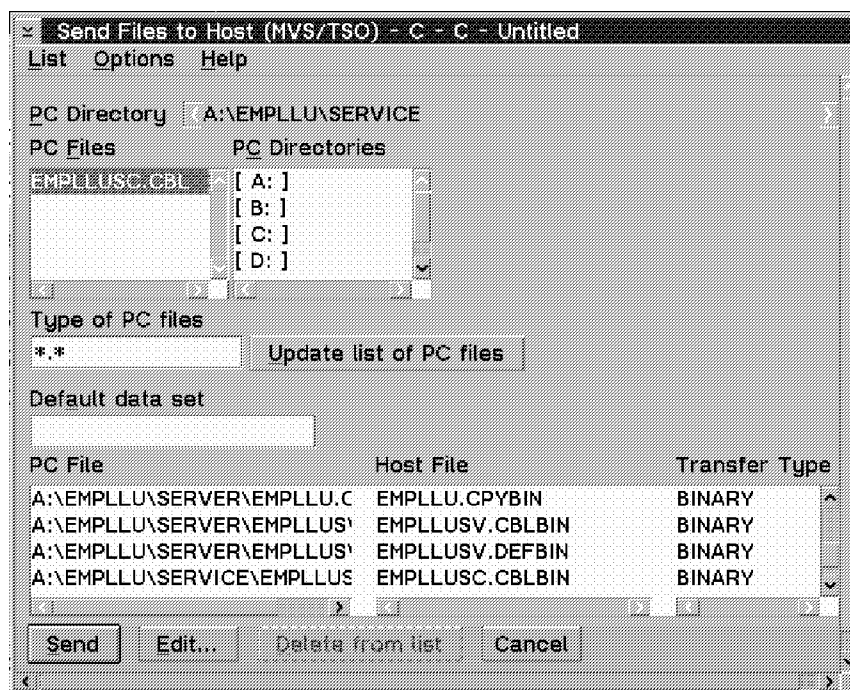


Figure 259. Send Files to Host (MVS/TSO) Window—Transfer File List

11. The list for the file transfer is complete. Now change the host file names and the transfer types for each of these files. Go through the files, select each of them, and click on **Edit...** The Edit Send List (MVS/TSO) window appears (Figure 260). Change the **Transfer type** from the default **BINARY** into **TEXT** and change the entry in the **Host file** to the appropriate data set name and file name for your environment. Click on **Change list** to update the file name in the transfer list.

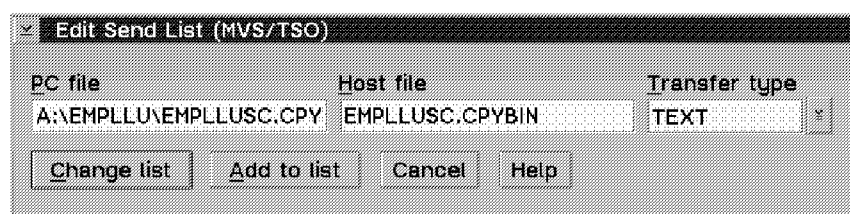


Figure 260. Edit Send List (MVS/TSO) Window

12. After finishing the update of all files, click on **Send** on the Send Files to Host window.
13. The Send File(s) Status window appears (Figure 261 on page 192) and shows the status of the files that will be sent. Make sure that you transfer seven files.

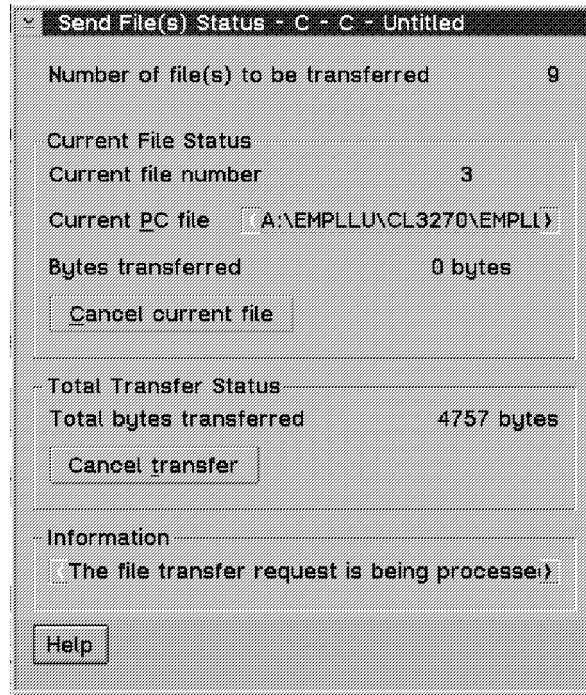


Figure 261. Send File(s) Status Window

14. After the transfer has finished, the Send File(s) Status window disappears. Click on **Cancel** on the Send Files to Host window to close this window. An OS/2 Communications Manager message window appears and asks you if you want to save the transfer list or not. Click on **Discard** (Figure 262) to leave the transfer function without saving the file list.

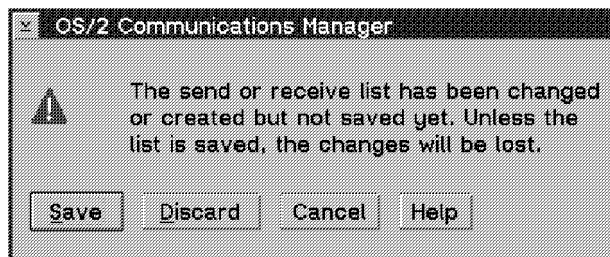


Figure 262. OS/2 Communications Manager Window—Save List

15. On the emulation window, you can verify the receipt of the files in the list shown below (Figure 263 on page 193).

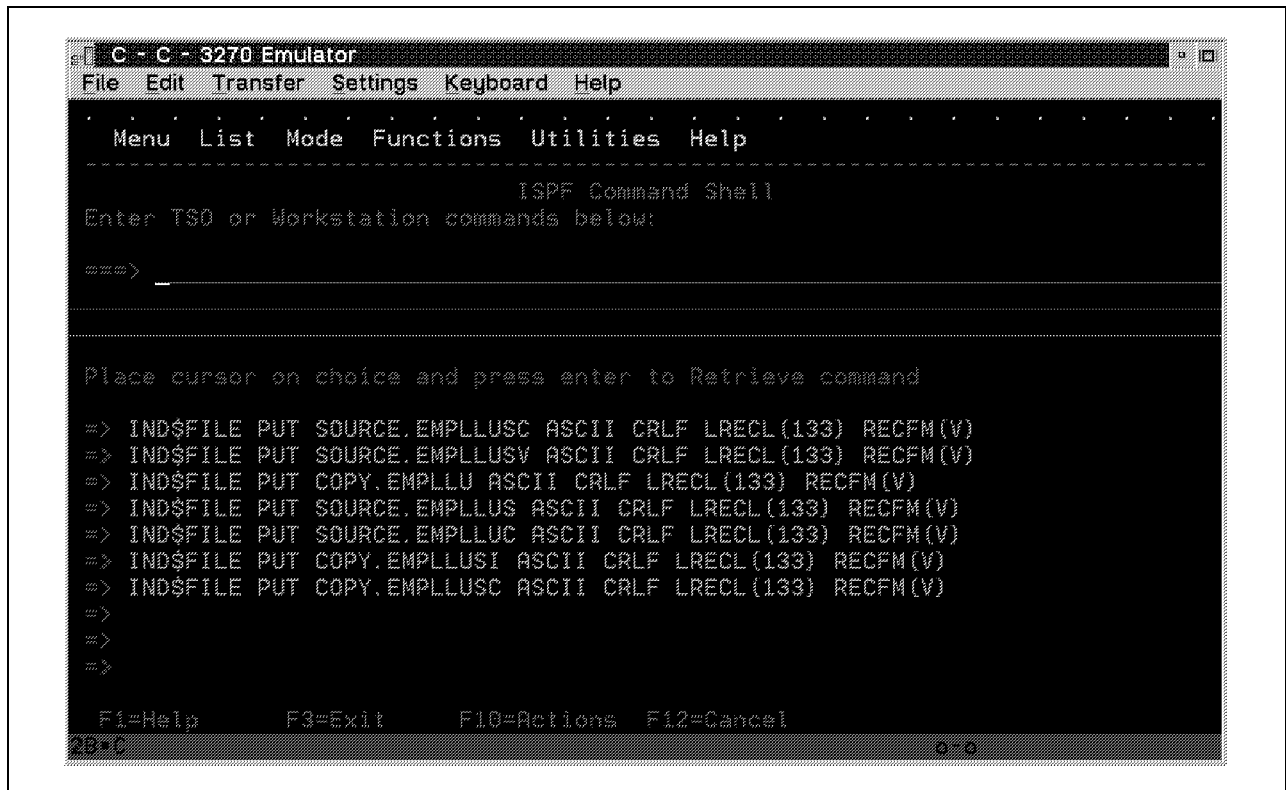


Figure 263. C-C-3270 Emulator Window—ISPF Command Shell

The file transfer is finished.

5.5 Prepare the Application on the MVS System

Your application source code is now back on the MVS system. In order to run the transaction in your CICS/ESA environment, you first need to do the following:

- Make the necessary changes in the source.
- Run all the jobs to create the load modules.
- Define the transaction, programs, and maps in CICS/ESA.

Note: The user ID we use on our MVS system is **MVSRES1**. All the user datasets have this user ID as the first-level qualifier, and the job names in the sample JCLs start with it.

You probably have to use a different user ID in your installation. Therefore you have to substitute your user ID for MVSRES1 wherever MVSRES1 is referenced in the following samples.

You have already created the following datasets in order to upload the source code to:

MVSRES1.CSCB.SOURCE	COBOL source code and BMS definition
MVSRES1.CSCB.COPY	COBOL copybooks

The jobs you are going to run need additional datasets. You will need to allocate the following datasets:

MVSRES1.CSCB.JCL	Job control language
-------------------------	----------------------

MVSRES1.CSCB.OBJ	Object decks
MVSRES1.CSCB.LOAD	Load modules
MVSRES1.CSCB.DBRM	Data base request module (output of DB2 preprocessor, input for DB2 bind).

5.5.1 Adapt the Source Code for MVS

In our sample application, four files are different on OS/2 and on MVS:

- The BMS map definition needs to be changed because of the EBCDIC character set used on the host.
- The copybook for the communication area between the client and the server program (EMPLLUSI) contains a statement that is valid only on the workstation. Adjustment needs to be done.
- The server program has to be changed to work with the DB2 for MVS precompiler.
- The copybook containing the DB2 table definition and its structure has to be created.

5.5.1.1 Change the Map Definition

Edit member EMPLLUS of the MVSRES1.CSCB.SOURCE dataset.

Change the value for the XINIT parameter to 40, the hex code for the blank character in the EBCDIC character set.

```

:
*   EXIT MAP.
EMPLLEX DFHMDI SIZE=(24,80),CTRL=(FREEKB),
        COLUMN=1,LINE=1
        DFHMDF POS=(1,1),ATTRB=(UNPROT,NORM,IC),LENGTH=75,
        XINIT=40
:

```

The CICS translator on OS/2 allows blank lines in the BMS map source. The Assembler compiler on MVS doesn't allow blank lines in the source code. Make sure you add an asterisk on Column 1 of any blank line in your BMS source to identify the line as a comment.

5.5.1.2 Change the Communication Area Definition

Our sample application contains two copybooks that are used by more than one program: the parameter passed between the server program and the service calculation routine (EMPLLUSC) and the CICS communication area between the client and the server program (EMPLLUSI).

For the host offload projects (EMPLLU Host Client and EMPLLU Host Server) you did not download the two copybooks nor copy them again from the diskette. Instead, you used the same files as in the OS/2 LAN with CICS and DB2 projects. This is because if the two copybooks are working on MVS, they are also working on OS/2 without change.

But this is not true the other way round, which is why you need to make one change in the EMPLLUSI member to make it work on MVS.

The server program passes the data from the DB2 table to the client program using the CICS communication area. Therefore, on OS/2 we included in the

structure for the communication area (EMPLLUSI) the structure for the DB2 table generated by the Data Assistant (EMPLLU.CPY). The Data Assistant generates for the length variable of a variable-length field of COMP-5. Because COMP-5 is not supported on MVS you have to substitute COMP for COMP-5 in the corresponding definitions of the EMPLLUSI member.

5.5.1.3 Change the Server Program Source

To make the server program run on MVS, you have to undo the changes you made in this program on the workstation:

- Change the FROM clause in the DECLARE CSR1 CURSOR statement to:

```
FROM DSN8410.EMP
```

This is because the table names of the sample database are not all the same on MVS as on OS/2.

If you don't use synonyms you must fully qualify the table name because you cannot log on to DB2 for MVS with the user ID that created the sample database as you do on OS/2. DSN8410 might not be the TBCREATOR for the sample tables in your installation. Check with your DB administrator.

- At the beginning of the 1000-MAIN section, delete the following statement:

```
EXEC SQL CONNECT TO SAMPLE END-EXEC.
```

- Change the first statement of the 1130-FETCH-ALL section to:

```
INITIALIZE DCLEMP.
```

This is the variable you will generate in the following section.

5.5.1.4 Create the Copybook containing the Table Definition

The EMPLLU copybook you created on OS/2 using the Data Assistant can be generated through the DCLGEN utility on MVS.

You cannot just upload the copybook Data Assistant generated because DB2 for MVS needs an additional declare statement that will also be created by the DCLGEN utility.

To generate the EMPLLU copy book on MVS, do the following:

1. Type the appropriate option for DB2I on the command line of your ISPF Primary Option Menu (Figure 264 on page 196) and press Enter. Ask your system administrator how to access DB2 Interactive if you don't have a corresponding option on your menu.

Menu Utilities Compilers Options Status Help			

ISPF Primary Option Menu			
Option ==> d			
0	Settings	Terminal and user parameters	< Calendar >
1	View	Display source data or listings	July 1996
2	Edit	Create or change source data	Su Mo Tu We Th Fr Sa
3	Utilities	Perform utility functions	1 2 3 4 5 6
4	Foreground	Interactive language processing	7 8 9 10 11 12 13
5	Batch	Submit job for language processing	14 15 16 17 18 19 20
6	Command	Enter TSO or Workstation commands	21 22 23 24 25 26 27
7	Dialog Test	Perform dialog testing	28 29 30 31
8	LM Facility	Library administrator functions	
9	IBM Products	IBM program development products	Time . . . : 08:38
10	SCLM	SW Configuration Library Manager	Day of year. : 197
D	DB2I	Access to DB2 V410	
Q	SDSF	System Display and Search Facility	
R	RACF	Resource Access Control Facility	
T	TUTORIAL	Display information about ISPF/PDF	
Enter X to Terminate using log/list defaults			

Figure 264. ISPF Primary Option Menu

- The DB2I Primary Option Menu appears (Figure 265 on page 197). The *SSID* field in the upper right corner shows you the name of the DB2 subsystem you are working with. In our installation it is **DB41**.
Type 2 on the command line and press Enter.

```
DB2I PRIMARY OPTION MENU                      SSID: DB41
COMMAND ==> 2

Select one of the following DB2 functions and press ENTER.

1  SPUFI                      (Process SQL statements)
2  DCLGEN                     (Generate SQL and source language declarations)
3  PROGRAM PREPARATION        (Prepare a DB2 application program to run)
4  PRECOMPILE                 (Invoke DB2 precompiler)
5  BIND/REBIND/FREE           (BIND, REBIND, or FREE plans or packages)
6  RUN                        (RUN an SQL program)
7  DB2 COMMANDS               (Issue DB2 commands)
8  UTILITIES                  (Invoke DB2 utilities)
D  DB2I DEFAULTS              (Set global parameters)
X  EXIT                       (Leave DB2I)

PRESS:                          END to exit      HELP for more information
```

Figure 265. DB2I PRIMARY OPTION MENU

3. The DCLGEN panel appears (Figure 266 on page 198).

Type EMP in the *SOURCE TABLE NAME* field, and DSN8410 in the *TABLE OWNER* field or whatever is used as TBCREATOR for the sample tables in your installation. Type 'MVSRES1.CSCB.COPY(EMPLLU)' in the *DATA SET NAME* field. Specify REPLACE in the *ACTION* field. If the member EMPLLU doesn't yet exist in the corresponding dataset, it will be added; otherwise it will be replaced.

Be sure that the DB2 subsystem you are using is started, and press Enter.

DCLGEN		SSID: DB41
===>		
Enter table name for which declarations are required:		
1	SOURCE TABLE NAME ===> EMP	(Unqualified)
2	TABLE OWNER ===> DSN8410	(Optional)
3	AT LOCATION ===>	(Optional)
Enter destination data set: (Can be sequential or partitioned)		
4	DATA SET NAME ... ===> 'MVSRES1.CSCB.COPY(EMPLLU) '	
5	DATA SET PASSWORD ===>	(If password protected)
Enter options as desired:		
6	ACTION ===> REPLACE	(ADD new or REPLACE old declaration)
7	COLUMN LABEL ===> NO	(Enter YES for column label)
8	STRUCTURE NAME .. ===>	(Optional)
9	FIELD NAME PREFIX ===>	(Optional)
10	DELIMIT DBCS ===> YES	(Enter YES to delimit DBCS identifiers)
11	COLUMN SUFFIX ... ===> NO	(Enter YES to append column name)
12	INDICATOR VARS .. ===> NO	(Enter YES for indicator variables)
PRESS: ENTER to process END to exit HELP for more information		

Figure 266. DCLGEN Panel

4. The following message is displayed:

```
DSNE905I EXECUTION COMPLETE, MEMBER EMPLLU ADDED
***
```

Press Enter.

5.5.2 Build the MVS Application

On MVS, you have to run several jobs in order to create the following components:

- BMS map symbolic description (copy book)
- Physical BMS map
- Client program load module
- Service calculation routine object deck
- Server program load module

Note: In the following sections, you will find examples of the corresponding job control members. Ask your system administrator about the accounting information, job class, and message class you need to use in your installation.

Authorizations: Make sure you have at least read access to all the datasets specified in these jobs and BINDADD authority to bind the application plan in DB2.

5.5.2.1 Create the BMS Maps for CICS for MVS/ESA

You can create the BMS map symbolic description and the physical map in one job.

In the BMS source (EMPLUS), we specified `TYPE=&SYSPARM` in the `DFHMSD` macro. The `CICSMAP` command on OS/2 runs the translator twice with the different values for the `SYSPARM` parameter. On MVS, you have to put three steps into your job:

- Create symbolic description with `SYSPARM(DSECT)`.
- Assemble physical maps with `SYSPARM(MAP)`.
- Link physical maps.

`.figref refid=mohs10.` is an example of the job control language (JCL) used to create the BMS maps:

```
//MVSRES1A JOB (999,POK), 'CICSUSR', NOTIFY=&SYSUID,
// CLASS=A, MSGCLASS=T, REGION=6M, MSGLEVEL=(1,1)
/*
/** STEP1 - CREATE CICS LOGICAL MAP
//MAPLA EXEC PGM=IEV90,
// REGION=2M, PARM='DECK,NOOBJECT,SYSPARM(DSECT)'
//SYSLIB DD DSN=CICS.V4R1M0.SDFHMAC, DISP=SHR
// DD DSN=CICS.V4R1M0.SDFHSRC, DISP=SHR
// DD DSN=SYS1.MACLIB, DISP=SHR
// DD DSN=SYS1.MODGEN, DISP=SHR
//SYSUT1 DD UNIT=SYSDA, SPACE=(1700,(400,400))
//SYSUT2 DD UNIT=SYSDA, SPACE=(1700,(400,400))
//SYSUT3 DD UNIT=SYSDA, SPACE=(1700,(400,400))
//SYSPRINT DD SYSOUT=*
//SYSIN DD DISP=SHR, DSN=MVSRES1.CSCB.SOURCE(EMPLUS) <== BMS source
//SYSPUNCH DD DISP=OLD, DSN=MVSRES1.CSCB.COPY(EMPLUS) <== copy structure
/*
/** STEP2 - CREATE CICS PHYSICAL MAP
//MAPPA EXEC PGM=IEV90, COND=(4,LT),
// REGION=2M, PARM='SYSPARM(MAP)'
//SYSLIB DD DSN=CICS.V4R1M0.SDFHMAC, DISP=SHR
// DD DSN=CICS.V4R1M0.SDFHSRC, DISP=SHR
// DD DSN=SYS1.MACLIB, DISP=SHR
// DD DSN=SYS1.MODGEN, DISP=SHR
//SYSUT1 DD UNIT=SYSDA, SPACE=(1700,(400,400))
//SYSUT2 DD UNIT=SYSDA, SPACE=(1700,(400,400))
//SYSUT3 DD UNIT=SYSDA, SPACE=(1700,(400,400))
//SYSPRINT DD SYSOUT=*
//SYSIN DD DISP=SHR, DSN=MVSRES1.CSCB.SOURCE(EMPLUS) <== BMS source
//SYSPUNCH DD DSN=&&TEMP, UNIT=SYSDA, DISP=(,PASS),
// DCB=(RECFM=F, BLKSIZE=80), SPACE=(1024,(100,10))
```

Figure 267 (Part 1 of 2). Job Control Language for Create BMS Sample

```

/*
/* *   STEP3 - LINKEDIT CICS PHYSICAL MAP
//MAPPL  EXEC PGM=IEWL,PARM='LIST,LET,XREF',COND=(4,LT)
//SYSUT1  DD UNIT=SYSDA,SPACE=(1024,(100,10))
//SYSLMOD DD DISP=SHR,DSN=MVSRES1.CSCB.LOAD           <== output dataset
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSLIN  DD DSN=&&TEMP,DISP=(OLD,DELETE)
//          DD *
NAME EMPLUS(R)                                     <== map set name

```

Figure 267 (Part 2 of 2). Job Control Language for Create BMS Sample

After the successful run of this job, copy the member EMPLUS from the specified output dataset (MVSRES1.CSCB.LOAD) to a dataset that is allocated to the DFHRPL ddname of the CICS startup job. In our installation we use CICS.V4R1M0.USER.LOADLIB for this purpose.

5.5.2.2 Compile and Link the Client Program

The job to compile and link the client program consists of three steps:

- CICS preprocessing
- Compile
- Link

Make sure you run the BMS creation job first because the client program includes the map structure, which is created by the BMS job.

Here is an example of the JCL used to compile and link the client program:

```

//MVSRES1B JOB (999,POK), 'CICSUSR',NOTIFY=&SYSUID,
// CLASS=A,MSGCLASS=T,REGION=6M,MSGLEVEL=(1,1)
/*
//TRN      EXEC PGM=DFHECP1$,
//          PARM=('COBOL2'),
//          REGION=2M
//STEPLIB  DD DSN=CICS.V4R1M0.SDFHLOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DSN=MVSRES1.CSCB.SOURCE(EMPLLUC),DISP=SHR   <== COBOL source
//SYSPUNCH DD DSN=&&SYSCIN,
//          DISP=(,PASS),UNIT=SYSDA,
//          DCB=BLKSIZE=400,
//          SPACE=(400,(400,100))

```

Figure 268 (Part 1 of 2). Job Control Language to Compile and Link the Client Program

```

/*
//COB      EXEC PGM=IGYCRCTL,REGION=2M,
//          PARM='NODYNAM,LIB,OBJECT,RENT,APOST,LIST,MAP,XREF'
//STEPLIB DD DSN=IGY.V1R2M0.SIGYCOMP,DISP=SHR
//SYSLIB  DD DSN=CICS.V4R1M0.SDFHCOB,DISP=SHR
//          DD DSN=CICS.V4R1M0.SDFHMAC,DISP=SHR
//          DD DSN=CICS.V4R1M0.SDFHSAMP,DISP=SHR
//          DD DSN=MVSRES1.CSCB.COPY,DISP=SHR                <== copy books
//SYSPRINT DD SYSOUT=*
//SYSIN   DD DSN=&&SYSCIN,DISP=(OLD,DELETE)
//SYSLIN  DD DSN=&&LOADSET,DISP=(MOD,PASS),
//          UNIT=SYSDA,SPACE=(80,(250,100))
//SYSUT1  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT2  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT3  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT4  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT5  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT6  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT7  DD UNIT=SYSDA,SPACE=(460,(350,100))

/*
//LKED     EXEC PGM=IEWL,REGION=2M,
//          PARM=('XREF,LIST,MAP,LET,AMODE=31,RMODE=ANY'),
//          COND=(5,LT,COB)
//SYSLIB  DD DSN=CICS.V4R1M0.SDFHLOAD,DISP=SHR
//          DD DSN=CEE.V1R5M0.SCEELKED,DISP=SHR
//SYSLMOD DD DSN=MVSRES1.CSCB.LOAD,DISP=OLD                <== output dataset
//SYSUT1  DD UNIT=SYSDA,DCB=BLKSIZE=1024,
//          SPACE=(1024,(200,20))
//SYSPRINT DD SYSOUT=*
//OBJECT  DD DSN=&&LOADSET,DISP=(OLD,DELETE)
//CICSOB  DD DSN=CICS.V4R1M0.SDFHCOB,DISP=SHR
//SYSIN   DD DUMMY
//SYSLIN  DD *
//          INCLUDE CICSOB(DFHEILIC)
//          INCLUDE OBJECT
//          NAME      EMPLLUC(R)                            <== client program

```

Figure 268 (Part 2 of 2). Job Control Language to Compile and Link the Client Program

After the successful run of this job, you need to copy the member EMPLLUC from the specified output dataset (MVSRES1.CSCB.LOAD) to a dataset which is allocated to the DFHRPL ddname of the CICS startup job. In our installation, we use CICS.V4R1M0.USER.LOADLIB for this purpose.

5.5.2.3 Compile the Service Calculation Routine

The job to compile the service calculation routine consists of only a compile step. There is no preprocessing and it does not have to be linked. The service calculation routine will be linked statically to the server program.

This step could be included in the compile/link job of the server program. The object deck could then be written to a temporary dataset, which will be deleted at the end of the job. But if it would be a standard routine used by several programs, you would not want to have to recompile it every time. Therefore, we create a separate routine for this compile operation and keep the object module.

Here is an example of the JCL used to compile the service calculation routine:

```

//MVSRES1C JOB (999,POK) , 'CICSUSR' ,NOTIFY=&SYSUID,
// CLASS=A,MSGCLASS=T,REGION=6M,MSGLEVEL=(1,1)
/*
//COB      EXEC PGM=IGYCRCTL,REGION=2M,
//          PARM='NODYNAM,LIB,OBJECT,RENT,APOST,LIST,MAP,XREF'
//STEPLIB DD DSN=IGY.V1R2M0.SIGYCOMP,DISP=SHR
//SYSLIB  DD DSN=MVSRES1.CSCB.COPY,DISP=SHR           <== copy books
//SYSPRINT DD SYSOUT=*
//SYSIN   DD DSN=MVSRES1.CSCB.SOURCE(EMPLLUSC),DISP=SHR   <== COBOL source
//SYSLIN  DD DSN=MVSRES1.CSCB.OBJ(EMPLLUSC),DISP=OLD      <== output member
//SYSUT1  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT2  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT3  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT4  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT5  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT6  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT7  DD UNIT=SYSDA,SPACE=(460,(350,100))

```

Figure 269. Job Control Language to Compile the Service Calculation Routine

Submit this job before you go on to the next section.

5.5.2.4 Compile, Link, and Bind the Server Program

The job to compile and link the client program consists of five steps:

- DB2 preprocessing
- CICS preprocessing
- Compile
- Link
- DB2 Bind

Make sure you run the service calculation routine compile job first because the object deck created by this job will be linked to the server program.

The EMPLLUSC structure that passes the parameters to the Service Calculation routine is based on the USA date format (mm/dd/yyyy). Therefore, if your DB2 for MVS has been installed with a different date format, you need to specify DATE(USA) as parameter for the DB2 precompile program (DSNHPC).

If you downloaded the application from the host, you would have had to specify the corresponding option on OS/2, in case of different date formats.

.figref refid=mohs13. is an example of the JCL used to compile, link, and bind the server program.

```

//MVSRES1D JOB (999,POK) , 'CICSUSR',NOTIFY=&SYSUID,
// CLASS=A,MSGCLASS=T,REGION=6M,MSGLEVEL=(1,1)
/*
//DBPRE EXEC PGM=DSNHPC,
//          PARM='HOST(COB2),XREF,SOURCE,FLAG(I),APOST,DATE(USA)'
//STEPLIB DD DSN=DSN410.SDSNEXIT,DISP=SHR
//          DD DSN=DSN410.SDSNLOAD,DISP=SHR
//DBRMLIB DD DSN=MVSRES1.CSCB.DBRM(EMPLLUSV),DISP=SHR          <== DBRM member
//SYSCIN  DD DSN=&&DSNHOUT,DISP=(MOD,PASS),UNIT=SYSDA,
//          SPACE=(800,(500,500))
//SYSIN   DD DSN=MVSRES1.CSCB.SOURCE(EMPLLUSV),DISP=SHR        <== COBOL source
//SYSLIB  DD DSN=MVSRES1.CSCB.COPY,DISP=SHR                    <== copy books
//SYSPRINT DD SYSOUT=*
//SYSTEM  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSUT1  DD UNIT=SYSDA,SPACE=(800,(500,500),,,ROUND)
//SYSUT2  DD UNIT=SYSDA,SPACE=(800,(500,500),,,ROUND)

/*
//TRN     EXEC PGM=DFHECP1$,
//          PARM=('COBOL2'),
//          REGION=2M,COND=(4,LT,DBPRE)
//STEPLIB DD DSN=CICS.V4R1M0.SDFHLOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN   DD DSN=&&DSNHOUT,DISP=(OLD,DELETE)
//SYSPUNCH DD DSN=&&SYSCIN,
//          DISP=(,PASS),UNIT=SYSDA,
//          DCB=BLKSIZE=400,
//          SPACE=(400,(400,100))

/*
//COB     EXEC PGM=IGYCRCTL,REGION=2M,COND=(4,LT,TRN),
//          PARM='NODYNAM,LIB,OBJECT,RENT,APOST,LIST,MAP,XREF'
//STEPLIB DD DSN=IGY.V1R2M0.SIGYCOMP,DISP=SHR
//SYSLIB  DD DSN=CICS.V4R1M0.SDFHCOB,DISP=SHR
//          DD DSN=CICS.V4R1M0.SDFHMAC,DISP=SHR
//          DD DSN=CICS.V4R1M0.SDFHSAMP,DISP=SHR
//          DD DSN=MVSRES1.CSCB.COPY,DISP=SHR                    <== copy books
//SYSPRINT DD SYSOUT=*
//SYSIN   DD DSN=&&SYSCIN,DISP=(OLD,DELETE)
//SYSLIN  DD DSN=&&LOADSET,DISP=(MOD,PASS),
//          UNIT=SYSDA,SPACE=(80,(250,100))
//SYSUT1  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT2  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT3  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT4  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT5  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT6  DD UNIT=SYSDA,SPACE=(460,(350,100))
//SYSUT7  DD UNIT=SYSDA,SPACE=(460,(350,100))

```

Figure 270 (Part 1 of 2). Job Control Language to Compile, Link, and Bind the Server Program

```

/*
//LKED EXEC PGM=IEWL,REGION=2M,
//      PARM=( 'XREF,LIST,MAP,LET,AMODE=31,RMODE=ANY' ),
//      COND=(5,LT,COB)
//SYSLIB DD DSN=CICS.V4R1M0.SDFHLOAD,DISP=SHR
//      DD DSN=CEE.V1R5M0.SCEELKED,DISP=SHR
//      DD DSN=DSN410.SDSNLOAD,DISP=SHR
//SYSLMOD DD DSN=MVSRES1.CSCB.LOAD,DISP=OLD           <== output dataset
//SYSUT1 DD UNIT=SYSDA,DCB=BLKSIZE=1024,
//      SPACE=(1024,(200,20))
//SYSPRINT DD SYSOUT=*
//OBJECT DD DSN=&&LOADSET,DISP=(OLD,DELETE)
//EMPLSC DD DSN=MVSRES1.CSCB.OBJ,DISP=SHR
//CICSOB DD DSN=CICS.V4R1M0.SDFHCOB,DISP=SHR
//CICSDB DD DSN=CICS.V4R1M0.SDFHLOAD,DISP=SHR
//*ICSDB DD DSN=DSN410.SDSNLOAD,DISP=SHR
//SYSIN DD DUMMY
//SYSLIN DD *
//      INCLUDE CICSOB(DFHEILIC)
//      INCLUDE OBJECT
//      INCLUDE EMPLSC(EMPLLUSC)           <== serv calc rout
//      INCLUDE CICSDB(DSNCLI)
//      NAME EMPLLUSV(R)                  <== server program
/*
//BIND EXEC PGM=IKJEFT01,COND=(4,LT)
//STEPLIB DD DSN=DSN410.SDSNEXIT,DISP=SHR
//      DD DSN=DSN410.SDSNLOAD,DISP=SHR
//DBRMLIB DD DSN=MVSRES1.CSCB.DBRM,DISP=SHR           <== DBRM dataset
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
//      DSN SYSTEM(DB41)                   <== DB2 subsystem
//      BIND PLAN(EMPLLUSV) MEMBER(EMPLLUSV) +   <== DBRM member
//      ACTION(REPLACE) ISOLATION(CS)
END

```

Figure 270 (Part 2 of 2). Job Control Language to Compile, Link, and Bind the Server Program

After the successful run of this job you have to copy the member EMPLLUSV from the specified output dataset (MVSRES1.CSCB.LOAD) to a dataset which is allocated to the DFHRPL ddname of the CICS startup job. In our installation, we use CICS.V4R1M0.USER.LOADLIB for this purpose.

5.5.3 CICS for MVS/ESA Definitions

Unlike the situation on the workstation, you have to define all the transactions, programs, and maps of your application in CICS for MVS/ESA before you can run the transaction. You can do this online through the CEDA transaction:

1. On the VAMP screen, type the application name of the CICS region you are using and press Enter.
2. Log on with your user ID and password if you are prompted to do so. If not, just press the Pause key to clear the screen.
3. Type the following command:

```
CEDA DEF TRANS
```

and press Enter.

The Define Transaction panel appears (Figure 271).

Type EMPL in the *TRANSACTION* field, CSCB in the *Group* field, EMPLLLUC in the *PROGRAM* field, and optionally a description in the *DESCRIPTION* field.

Press Enter. The following messages are displayed at the bottom of the panel:

I New group CSCB created.

DEFINE SUCCESSFUL

DEF TRANS		
OVERTYPE TO MODIFY		CICS RELEASE = 0410
CEDA DEFINE TRANSACTION()		
TRANSACTION ==> EMPL		
Group ==> CSCB		
DESCRIPTION ==> EMPLOYEE LOOKUP		
PROGRAM ==> EMPLLLUC		
TWASIZE ==> 00000	0-32767	
PROFILE ==> DFHCICST		
PARTITIONSET ==>		
STATUS ==> Enabled	Enabled Disabled	
PRIMESIZE : 00000	0-65520	
TASKDATALOC ==> Below	Below Any	
TASKDATAKEY ==> User	User Cics	
STORAGECLEAR ==> No	No Yes	
RUNAWAY ==> System	System 0-2700000	
SHUTDOWN ==> Disabled	Disabled Enabled	
ISOLATE ==> Yes	Yes No	
REMOTE ATTRIBUTES		
+ DYNAMIC ==> No	No Yes	
MESSAGES: 2 SEVERE		
	SYSID=A	APPLID=SCMCICSA
PF 1 HELP 2 COM 3 END	6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF	12 CNCL

Figure 271. Define Transaction Panel

4. Press F3, type CEDA DEF PROG, and press Enter.

The Define Program panel appears (Figure 272 on page 206).

Type EMPLLLUC in the *PROGRAM* field, CSCB in the *Group* field, Le370 in the *Language* field, and optionally a description in the *Description* field.

Press Enter. The following message is displayed at the bottom of the panel:

DEFINE SUCCESSFUL

```

DEF PROG
OVERTYPE TO MODIFY
CEDA Define PROGRAM(
PROGRAM ==> EMPLLLUC
Group ==> CSCB
Description ==> EMPLOYEE LOOKUP CLIENT PROGRAM
Language ==> Le370 Cobol | Assembler | Le370 | C | Pli
| Rpg
RELoad ==> No No | Yes
RESident ==> No No | Yes
USAge ==> Normal Normal | Transient
USElpacopy ==> No No | Yes
Status ==> Enabled Enabled | Disabled
RSl : 00 0-24 | Public
Cedf ==> Yes Yes | No
Datalocation ==> Below Below | Any
EXECKey ==> User User | Cics
REMOTE ATTRIBUTES
REMOTESystem ==>
+ REMOTENAME ==>
MESSAGES: 2 SEVERE
SYSID=A APPLID=SCMCICSA
PF 1 HELP 2 COM 3 END 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Figure 272. Define Program Panel

5. Press F3, type CEDA DEF PROG again, and press Enter.

The Define Program panel appears (Figure 272).

Type EMPLPLUSV in the *PROG* field, CSCB in the *Group* field, Le370 in the *Language* field, and optionally a description in the *DEscription* field (for example, EMPLOYEE LOOKUP SERVER PROGRAM).

Press Enter. The following message is displayed at the bottom of the panel:

DEFINE SUCCESSFUL

6. Press F3, type CEDA DEF M, and press Enter.

The Define Mapset panel appears (Figure 273 on page 207).

Type **EMPLPLUS** in the *Mapset* field, **CSCB** in the *Group* field, and optionally a description in the *Description* field.

Press Enter. The following message is displayed at the bottom of the panel:

DEFINE SUCCESSFUL


```

DEF M
OVERTYPE TO MODIFY
CEDA DEFINE Mapset(
Mapset      ==> EMPLPLUS
Group       ==> CSCB
Description ==> EMPLOYEE LOOKUP MAPSET
Resident    ==> No           No | Yes
USAge       ==> Normal       Normal | Transient
USElpacopy  ==> No           No | Yes
Status      ==> Enabled      Enabled | Disabled
RS1         : 00             0-24 | Public

MESSAGES: 2 SEVERE

SYSID=A    APPLID=SCMCICSA

PF 1 HELP 2 COM 3 END      6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

7. Press F3 to exit the Define Mapset panel. All elements are now defined to CICS. In order to install them, type

and press Enter.

INSTALL SUCCESSFUL

```

I G(CSCB)
OVERTYPE TO MODIFY
CEDA Install
All
Connection ==>
File ==>
Lsrpool ==>
Mapset ==>
PARTitionset ==>
PARTNer ==>
PROFile ==>
PROGram ==>
TERminal ==>
TRANClass ==>
TRANSaction ==>
TYpeterm ==>
Group ==> CSCB

```

```

                                SYSID=A    APPLID=SCMCICSA
INSTALL SUCCESSFUL             TIME: 17.02.37 DATE: 96.255
PF 1 HELP      3 END          6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Figure 274. Install Panel

5.5.4 Run the Transaction on the Host

Before you can run the EMPL transaction, make sure that Language Environment support is installed in your CICS system. Ask your system administrator and refer to the corresponding CICS/ESA documentation.

Because the employee lookup server CICS program processes DB2 commands, the CICS-attachment facility provided by DB2 must be installed. Ask your system administrator how to install this DB2 support and refer to the corresponding DB2 for MVS documentation. The resource control table (RCT), which has to be generated for the definition of the DB2-CICS connection should contain the following statement:

```

.
.
.
DSNCRCT TYPE=ENTRY, TXID=EMPL, THRDM=1, THRDA=1, PLAN=EMPLLUSV, X
AUTH= (USERID, TERM, TXID)
.
.
.

```

Once the CICS-attachment facility is defined, you can start it using the following command in CICS:

```
DSNC STRT xx
```

where **xx** is the suffix of the RCT (the default is 00). Upon successful completion you get the following message:

```
DSN2023I THE ATTACHMENT FACILITY HAS CONNECTED TO 'DB41' USING 'DSN2CT00
```

To allow a CICS user to run the server program that accesses DB2 data, you have to provide the necessary authorization. You can use the SQL processor using file input (SPUFI) to execute the corresponding command:

1. In your TSO, go to the DB2I PRIMARY OPTION MENU as shown in Figure 265 on page 197 and select option 1.
2. The SPUFI panel appears (Figure 275). Type the name of a sequential dataset or a PDS member in the input *DATA SET NAME* field and the name of a sequential dataset name in the output *DATA SET NAME* field. The input dataset must be allocated, the output dataset will be allocated automatically if it does not exist. Press Enter.

SPUFI		SSID: DB41
====>		
Enter the input data set name: (Can be sequential or partitioned)		
1	DATA SET NAME ... ==>	CSCB.SOURCE(SPUFI)
2	VOLUME SERIAL ... ==>	(Enter if not cataloged)
3	DATA SET PASSWORD ==>	(Enter if password protected)
Enter the output data set name: (Must be a sequential data set)		
4	DATA SET NAME ... ==>	SPUFIOUT
Specify processing options:		
5	CHANGE DEFAULTS ==>	NO (Y/N - Display SPUFI defaults panel?)
6	EDIT INPUT ==>	YES (Y/N - Enter SQL statements?)
7	EXECUTE ==>	YES (Y/N - Execute SQL statements?)
8	AUTOCOMMIT ==>	YES (Y/N - Commit after successful run?)
9	BROWSE OUTPUT ... ==>	YES (Y/N - Browse output data set?)
For remote SQL processing:		
10	CONNECT LOCATION ==>	
PRESS: ENTER to process END to exit HELP for more information		

Figure 275. SPUFI Panel

3. The input dataset or member is shown in edit mode. Type the following statement in the edit area and press F3:

```
GRANT EXECUTE ON PLAN EMPLLVSV TO PUBLIC
```

The following message is displayed on the SPUFI panel:

```
DSNE808A EDIT SESSION HAS COMPLETED. PRESS ENTER TO CONTINUE
```

4. Press Enter again. The output dataset is displayed in browse mode. It contains the following messages:

```
-----+-----+-----+-----+-----+-----+-----+-----+
GRANT EXECUTE ON PLAN EMPLLVSV TO PUBLIC
-----+-----+-----+-----+-----+-----+-----+-----+
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
-----+-----+-----+-----+-----+-----+-----+-----+
DSNE617I COMMIT PERFORMED, SQLCODE IS 0
DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
-----+-----+-----+-----+-----+-----+-----+-----+
DSNE601I SQL STATEMENTS ASSUMED TO BE BETWEEN COLUMNS 1 AND 72
```

```
DSNE620I NUMBER OF SQL STATEMENTS PROCESSED IS 1
DSNE621I NUMBER OF INPUT RECORDS READ IS 1
DSNE622I NUMBER OF OUTPUT RECORDS WRITTEN IS 12
```

You are now ready to run the transaction. Type the transaction code `EMPL` on a cleared CICS screen and press Enter. The Employee Lookup panel appears (Figure 276).

Employee Lookup

Employee Name	Dept	Phone	Hire Dt	Svc Yrs
---------------	------	-------	---------	---------

Press "ENTER" to display or F3 to EXIT

Figure 276. Employee Lookup Panel on MVS

Press Enter. The Employee Lookup panel is displayed again showing the list of employees (Figure 277 on page 211).

Employee Lookup				
Employee Name	Dept	Phone	Hire Dt	Svc Yrs
ADAMSON, BRUCE .	D11	4510	1972-02-12	24
ALONZO, ROY R.	E21	5698	1947-05-05	49
BROWN, DAVID .	D11	4501	1966-03-03	30
GEYER, JOHN B.	E01	6789	1949-08-17	47
GOUNOT, JASON R.	E21	5698	1947-05-05	49
HAAS, CHRISTINE I.	A00	3978	1965-01-01	31
HEMMINGER, DIAN J.	A00	3978	1965-01-01	31
HENDERSON, EILEEN W.	E11	5498	1970-08-15	26
JEFFERSON, JAMES J.	D21	2094	1966-11-21	29
JOHN, REBA K.	D11	0672	1968-08-29	28
JOHNSON, SYBIL V.	D21	8953	1975-09-11	21
JONES, WILLIAM T.	D11	0942	1979-04-11	17
Press "ENTER" to display or F3 to EXIT				

Figure 277. Employee Lookup Panel—List of Employees on MVS

Chapter 6. OS/2 Client with MVS Server

In the previous chapters we described how to set up a workstation based client/server environment as well as a standalone MVS environment. This chapter combines these two environments which means that the GUI client program used in section 2.3.2.2, "Installing the Client Part of the Application" on page 95 calls the MVS server program used in section 5.3.2, "CICS-DB2 Server Application" on page 184.

To do this you don't have to change the COBOL programs or to install additional products. Just make sure you finished Chapter 2, "OS/2 LAN with CICS and DB2" on page 59 and Chapter 5, "Host Development Offload" on page 157.

The CICS Client product allows a non-CICS program to access directly the services of any CICS server using various communication protocols. Therefore our OS/2 based GUI client program does not have to access CICS for OS/2 as a gateway to the MVS CICS. However to communicate with a mainframe CICS only SNA is supported.

You have to define additional parameters in the following products in order to run the OS/2 - MVS client/server application:

- Communications Manager/2 for OS/2 Warp
- CICS Client for OS/2
- CICS for MVS/ESA
- VTAM

Figure 278 on page 214 shows an overview of the environment.

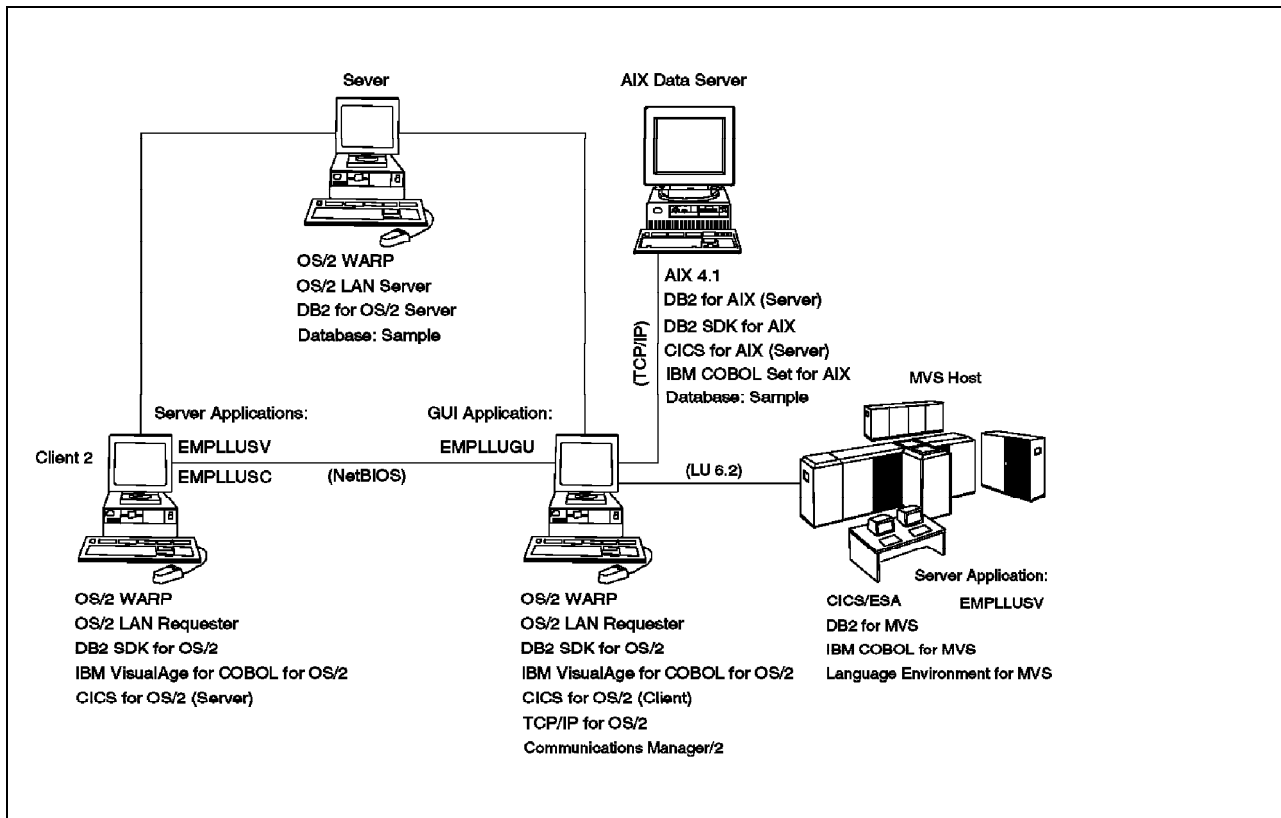


Figure 278. OS/2 Client with MVS Server

This environment is based on the setup described in Chapter 2, “OS/2 LAN with CICS and DB2” on page 59 and Chapter 5, “Host Development Offload” on page 157.

Host environment

Install Communications Manager/2 for OS/2 Warp on the workstation where your GUI client program is running as described in 5.1, “Installing and Configuring Communications Manager/2 for OS/2 Warp” on page 158 (3270 emulation). You can then use this workstation to log on to your MVS/ISPF and your CICS for MVS/ESA.

You might not be authorized to carry out all the VTAM and CICS definitions needed on your host system. Ask your system programmer to assist you.

Workstation setup

All you have to do on the workstation is to define the APPC (advanced program-to-program communication) link in Communications Manager/2 for OS/2 Warp, also known as LU6.2 or SNA.

6.1 Overview

In order to establish the communication between a workstation and a host system some parameters have to be defined first in various products. These products are:

On the workstation:

- Communications Manager/2 for OS/2 Warp

- CICS Client

On the MVS host:

- VTAM
- CICS for MVS/ESA

Some parameters are used in one product only, others are used in more than one or all the products. If this is the case they need to have a matching value.

The parameters that must match are shown in Table 1. Column five contains the values we used in our environment and the last column can be used to write down the valid values for your environment. We recommend that you complete this column first before you go on with the configuration of the four products as described in the next sections.

Some parameters might be defined already in your environment, for example the VTAM NETID or the CICS/ESA APPLID. Look for their values and fill in the corresponding cell of the table.

<i>Table 1. Matching Parameter Definitions</i>					
VTAM	CICS for MVS/ESA	Communications Manager/2 for OS/2 Warp	CICS Client	Our Example	Your Value
IDBLK/IDNUM	—	Local node ID	—	05D B1255	
LU	Netname	Local LU	LocalLUName	STB1255I	
LOGMODE	Modename	Mode name	ModeName	LU62APPB	
APPL	APPLID	Partner LU	Netname	SCMCICSA	
NETID	—	Network ID	Netname	USIBMSC	

Important: In the following sections we describe where and how these parameters have to be defined. Every time when we refer to a parameter value of the table above it is highlighted like **EXAMPLE**. You should then replace your value for our example.

6.2 Configuring Communications Manager/2 for OS/2 Warp

The following parameters have to be defined in Communications Manager/2 for OS/2 Warp:

- Matching with parameters of other products:
 - Local node ID
 - Local LU name
 - Mode name
 - Partner LU name
 - Network ID
- Additional definitions:
 - C&SM LAN ID
 - Local node name
 - LAN destination address

- Partner node name
- Change Number of Sessions (CNOS)

6.2.1 APPC Set-up

Follow these steps to set up APPC in Communications Manager/2 for OS/2 Warp:

1. On the desktop double-click on the **Communications Manager/2** icon. The Communications Manager/2—Icon View window appears as shown in Figure 229 on page 165.

If your Communications Manager is started, stop it by double-clicking on the **Stop Communications Normally** icon.

2. Double-click on the **Communications Manager Setup** icon. Click on **OK** on the Communications Manager/2 window (Figure 217 on page 159).
3. The Communications Manager Setup window is displayed as in Figure 219 on page 160. Click on **Setup....**
4. The Open Configuration window appears (Figure 279). If you installed Communications Manager/2 for OS/2 Warp as described in section 5.1, "Installing and Configuring Communications Manager/2 for OS/2 Warp" on page 158, *CSCBCM2* is already displayed in the *Configuration* field. Otherwise specify this value and optionally a description in order to create a new configuration file. Click on **OK**.

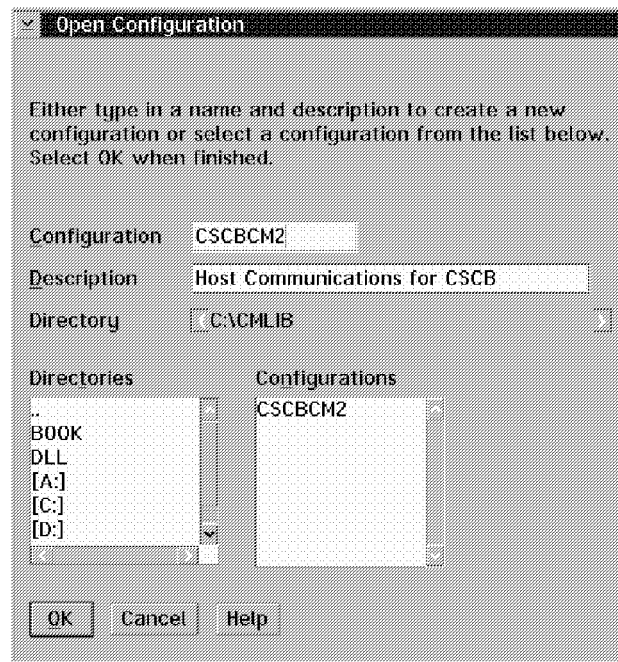


Figure 279. Open Configuration Window for APPC

5. The OS/2 Communications Manager window is displayed (Figure 280 on page 217). Click on **Yes**.

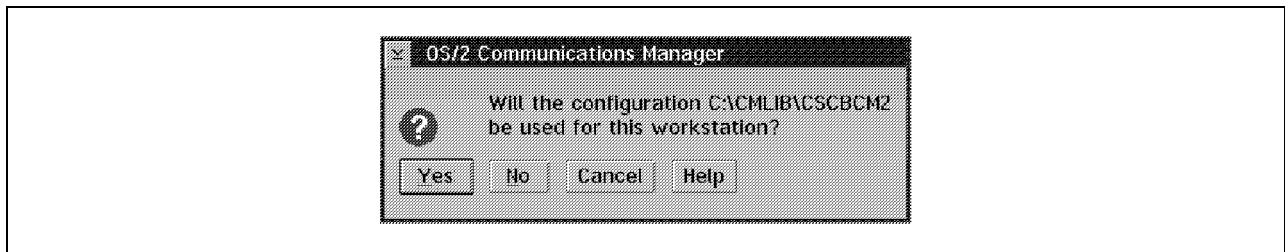


Figure 280. OS/2 Communications Manager Window for APPC

6. The Communications Manager Configuration Definition—CSCBCM2 window appears. Ensure that the **Additional definitions** radio button is selected.

Select **Options** from the menu bar, and **Use advanced configuration** from the first and **On** from the second pull-down menu.

Select **Token-ring or other LAN types** from the *Workstation Connection Type* list box and **APPC APIs** from the *Feature or Application* list box. The window shows a graphic of APPC APIs through Token-ring for communications (Figure 281).

Click on **Configure....**

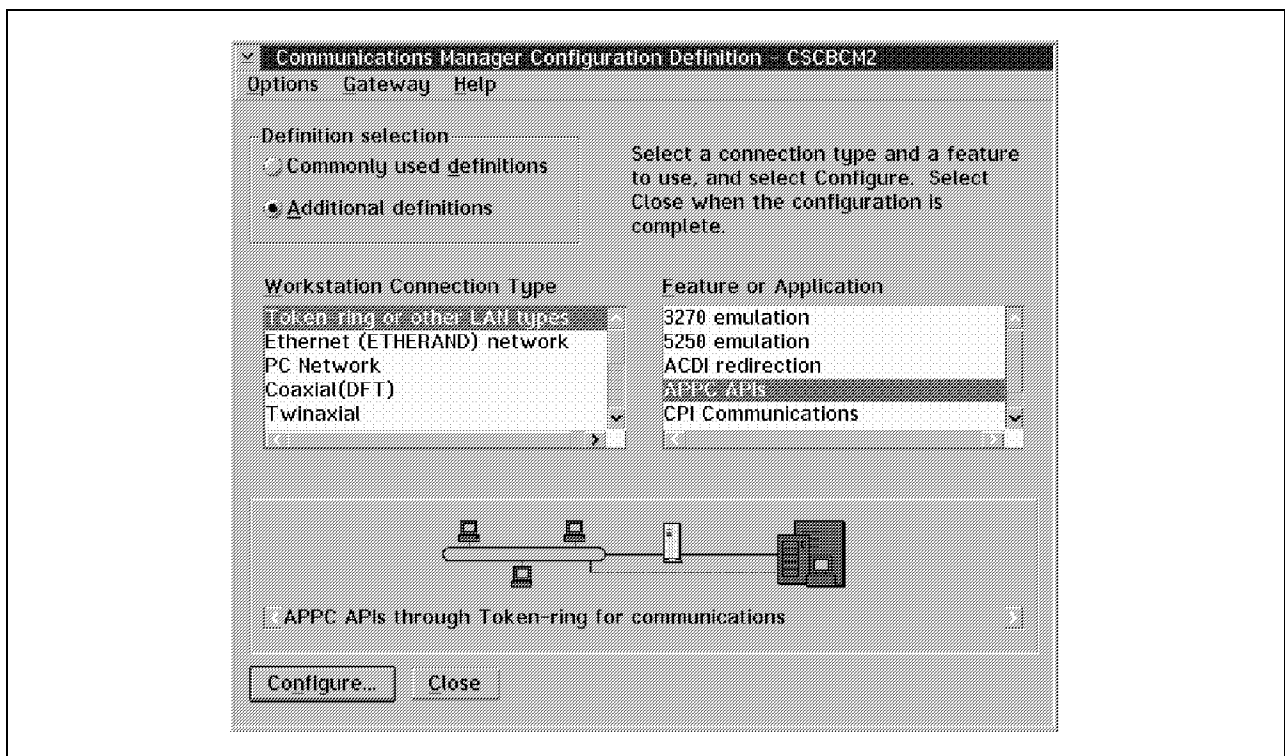


Figure 281. Communications Manager Configuration Definition—CSCBCM2 Window

7. The Communications Manager Profile List window appears (Figure 282 on page 218). Select **DLC—Token-ring or other LAN types** from the *Profile Name* list box and click on **Configure....**

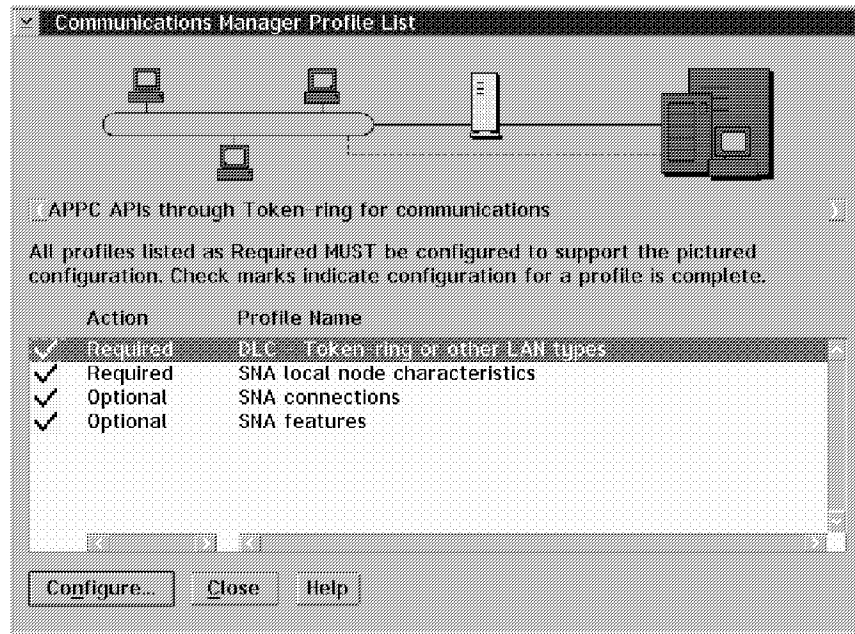


Figure 282. Communications Manager Profile List Window for APPC

8. Type the name of the network where the Token Ring is defined in the *C&SM LAN ID* field of the Token Ring or Other LAN Types DLC Adapter Parameters window (Figure 283). This field is used for the system management and has nothing to do with the APPC client/server set-up. Click on **OK**.

Figure 283. Token Ring or Other LAN Types DLC Adapter Parameters Window

9. Select **SNA local node characteristics** from the *Profile Name* list box of the Communications Manager Profile List Window (Figure 282) and click on **Configure....**

10. The Local Node Characteristics window is displayed (Figure 284).

Type your value for the Network ID in the *Network ID* field, in our example USIBMSC .

Type a name in the *Local node name* field. The local node name is the name that other nodes in the network use to address this node, for example, in an Advanced Peer-to-Peer Network (APPN). For our configuration this name is not relevant. But it has to be unique within the network. You can specify any meaningful name like the VTAM PU name as we used it.

Type your value for the Local node ID in the *Local node ID* field, in our example 05D B1255 .

Click on **OK**.

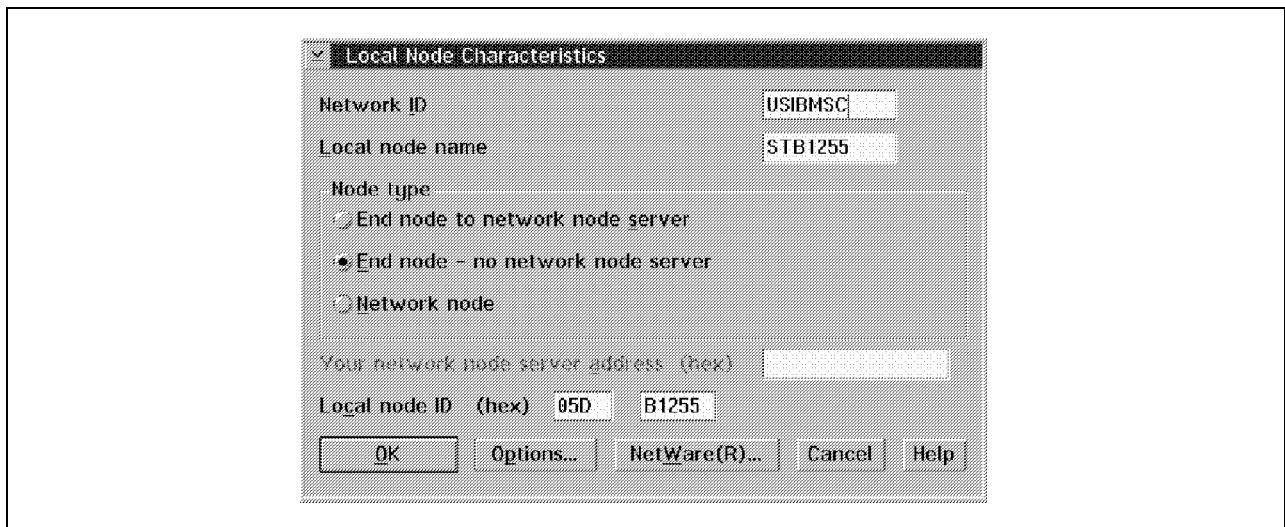


Figure 284. Local Node Characteristics Window

11. Select **SNA connections** from the *Profile Name* list box of the Communications Manager Profile List Window (Figure 282 on page 218) and click on **Configure....**

12. The Connections List window is displayed (Figure 285 on page 220).

Ensure that the **To host** radio button is selected and

If you have already defined a 3270 emulation for the current configuration the list box on the Connections List window shows the following entry:

HOST0001 Token-ring or other LAN types 0

If this is the case select this entry and click on **Change....** Otherwise click on **Create....**

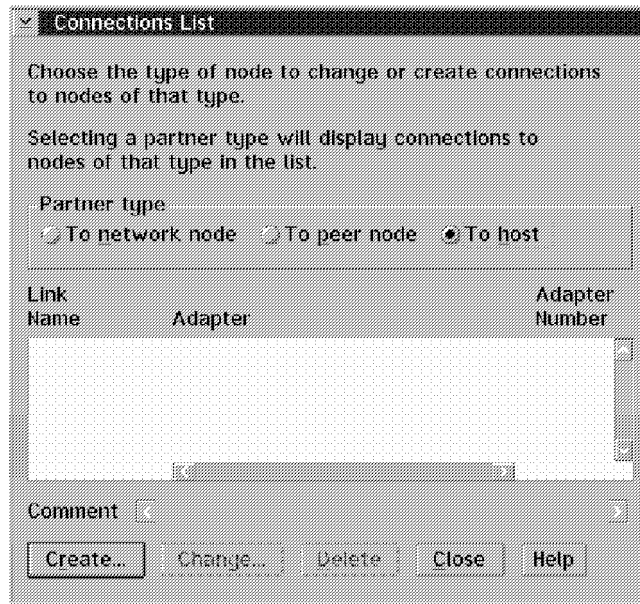


Figure 285. Connections List Window

13. Select **Token-ring or other LAN types** from the *Adapter Type* list box on the Adapter List window (Figure 286) and click on **Continue....**



Figure 286. Adapter List Window

14. The Connection to a Host window appears (Figure 287 on page 221).
Depending on if you are creating a new host link or changing an existing host link more or less fields are already filled.
Accept the default value for the *Link name* field (HOST0001) or replace a more meaningful name for it like we did (CSCBMVS). The link name is only known to your Communications Manager and is used to identify what parameter definitions belong together.
The Node ID should already be entered, in our example 05D B1255 .

Enter your LAN destination address in the *LAN destination address* field. In our network this is the token-ring address of the network controller.

Type your value for the Network ID in the *Partner network ID* field, in our example USIBMSC .

Type SCMCICSA in the *Partner node name* field.

You can type any comment in the *Optional comment* field or leave it blank.

Ensure that the **APPN support** check box is not checked.

Click on **Define Partner LUs...**

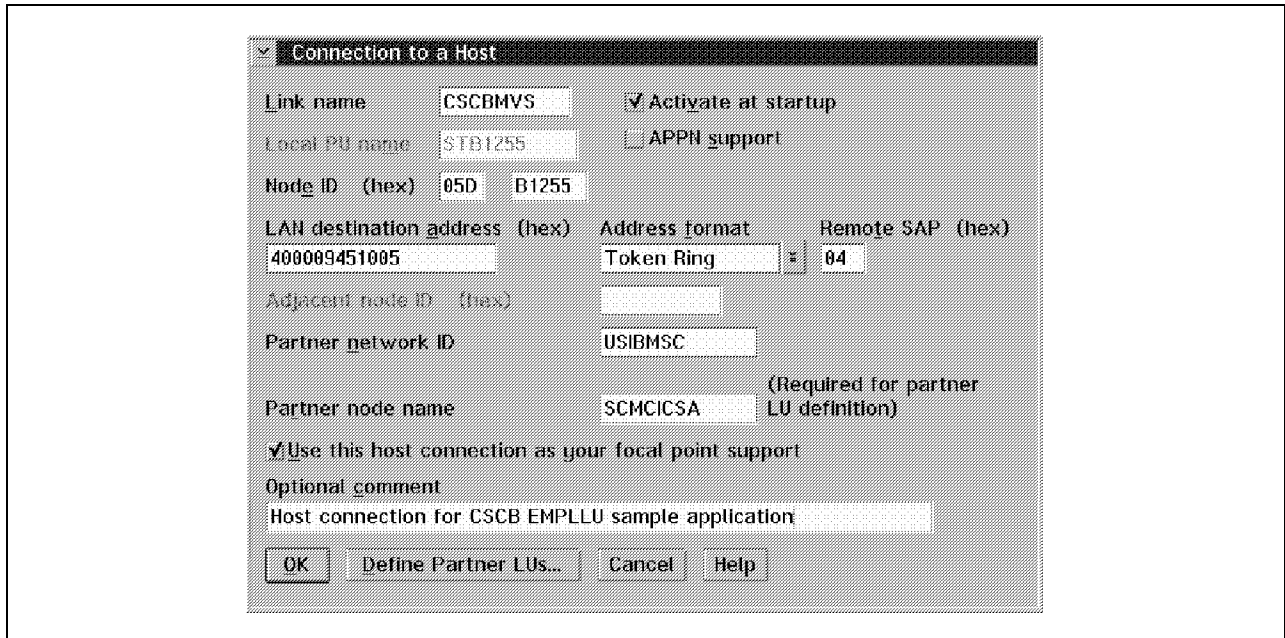


Figure 287. Connection to a Host Window

15. The Partner LUs window appears.

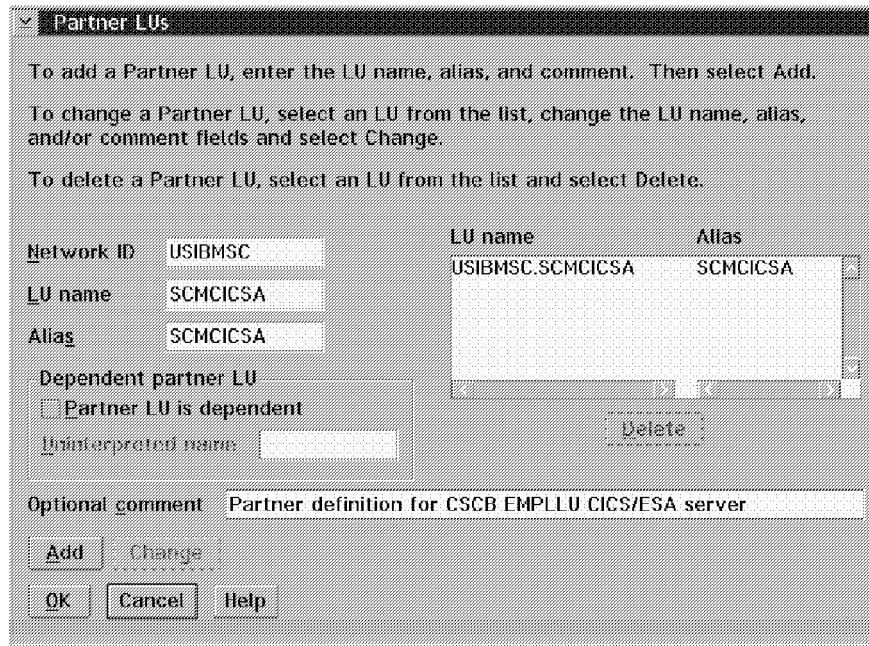
Type your value for the Network ID in the *Network ID* field, in our example USIBMSC .

Type your value for the Partner LU in the *LU name* and in the *Alias* field, in our example SCMCICSA .

You can type any comment in the *Optional comment* field or leave it blank.

Click on **Add**. The LU name is displayed in the *LU name* *Alias* list box (Figure 288 on page 222).

Click on **OK**.



Partner LUs

To add a Partner LU, enter the LU name, alias, and comment. Then select Add.

To change a Partner LU, select an LU from the list, change the LU name, alias, and/or comment fields and select Change.

To delete a Partner LU, select an LU from the list and select Delete.

Network ID: USIBMSC

LU name: SCMCICSA

Alias: SCMCICSA

Dependent partner LU

☐ Partner LU is dependent

Uninterpreted name:

LU name	Alias
USIBMSC.SCMCICSA	SCMCICSA

Optional comment: Partner definition for CSCB EMPLLU CICS/ESA server

Add Change

OK Cancel Help

Delete

Figure 288. Partner LUs Window

16. Click on **OK** on the Connection to a Host window.

If the OS/2 Communications Manager window appears as in Figure 289 click on **Change**.



OS/2 Communications Manager

 This connection has 3270 emulator definitions associated with it. If processing continues, the emulator definitions will also be changed.

Change Cancel Help

Figure 289. OS/2 Communications Manager Window (3270 emulator information)

Click on **Close** on the Connections List window which shows now the following entry in the list box:

CSCBMVS Token-ring or other LAN types 0

17. Select **SNA features** from the *Profile Name* list box of the Communications Manager Profile List Window (Figure 282 on page 218) and click on **Configure....**
18. The SNA Features List window is displayed (Figure 290 on page 223). Select **Local LUs** from the *Features* list box and click on **Create....**

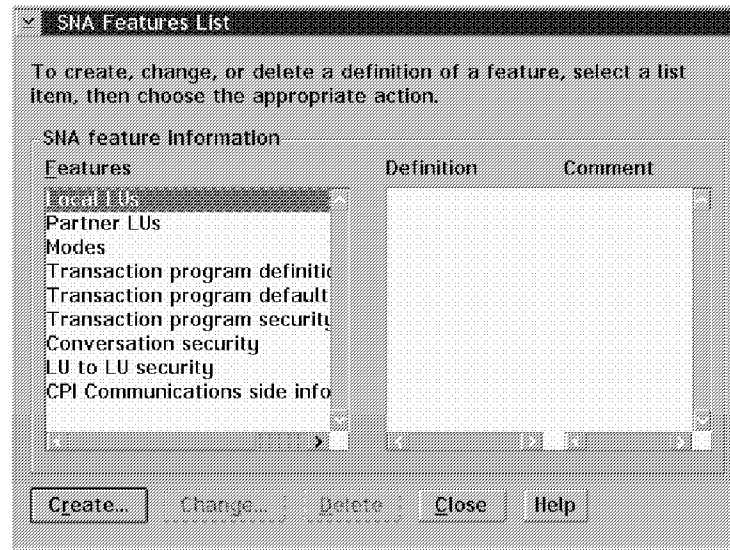


Figure 290. SNA Features List Window

19. The Local LU window is displayed (Figure 291).

Type your value for the Local LU in the *LU name* and in the *Alias* field, in our example STB1255I .

You can type any comment in the *Optional comment* field or leave it blank.

Click on **OK**.

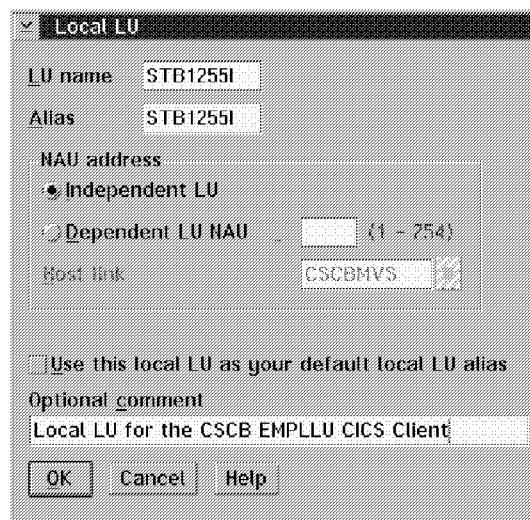


Figure 291. Local LU Window

20. The LU name is displayed in the *Definition* *Comment* list box of the SNA Features List window (Figure 290).

Select **Modes** from the *Features* list box and click on **Create....**

21. The Mode Definition window is displayed (Figure 292 on page 224).

Type your value for the Mode name in the *Mode name* field, in our example LU62APPB .

Select **#CONNECT** from the *Class of service* combination box.

You can type any comment in the *Optional comment* field or leave it blank.

Click on **OK**.

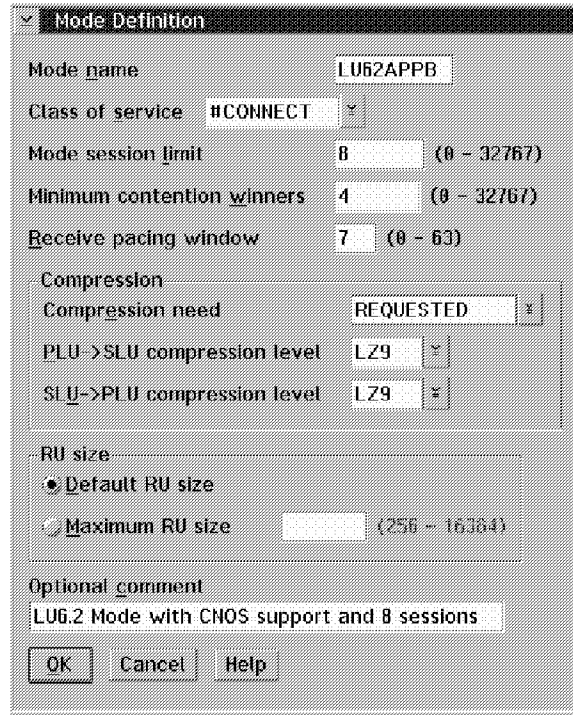
A screenshot of the 'Mode Definition' window. It contains several fields: 'Mode name' with the value 'LU62APPB', 'Class of service' with a dropdown menu showing '#CONNECT', 'Mode session limit' with a value of '8' and a range '(0 - 32767)', 'Minimum contention winners' with a value of '4' and a range '(0 - 32767)', and 'Receive pacing window' with a value of '7' and a range '(0 - 63)'. There is a 'Compression' section with 'Compression need' set to 'REQUESTED', 'PLU->SLU compression level' set to 'LZ9', and 'SLU->PLU compression level' set to 'LZ9'. Below this is an 'RU size' section with two radio buttons: 'Default RU size' (selected) and 'Maximum RU size' (with a range of '(256 - 16384)'). At the bottom is an 'Optional comment' field containing the text 'LU6.2 Mode with CNOS support and 8 sessions'. At the very bottom are 'OK', 'Cancel', and 'Help' buttons.

Figure 292. Mode Definition Window

22. Click on **Close** on the SNA Features List window.

Click on **Close** on the Communications Manager Profile List window.

Click on **Close** on the Communications Manager Configuration Definition—CSCBCM2 window.

The configuration is being validated while the Communications Manager—Checking Values window is shown (Figure 293).

A screenshot of the 'Communications Manager - Checking Values' window. It has a title bar with a dropdown arrow and the text 'Communications Manager - Checking Values'. The main area contains the text 'Please wait... checking configuration values.' in a monospaced font.

Figure 293. Communications Manager—Checking Values Window

Click on **Close** on the Communications Manager Setup window.

6.2.2 Change Number of Sessions

Before the client connection to the server can be started, the sessions between the client and the server must be started. This can be done using two methods:

- Every time when Communications Manager is started you can establish the LU 6.2 sessions through the Subsystem Management dialog of the Communications Manager.
- You can add the appropriate change number of sessions (CNOS) statements to the NDF file. This way the sessions will be established automatically each time when the Communications Manager is started. This is the preferred method.

To add the CNOS statements to your NDF file do the following:

1. Open the corresponding NDF file in edit mode.

The NDF files are located in the C:\CMLIB directory. The file is called like the configuration name you specified on the Open Configuration window (Figure 279 on page 216). In our example it is called CSCBCM2.NDF.

2. Add the statements shown in Figure 294 at the end of this file (after the START_ATTACH_MANAGER; statement) and save the NDF file.

```
:
CNOS  LOCAL_LU_ALIAS(STB1255I)
      FQ_PARTNER_LU_NAME( USIBMSC . SCMCICSA )
      MODE_NAME( LU62APPB )
      SET_NEGOTIABLE(NO)
      PLU_MODE_SESSION_LIMIT(8)
      MIN_CONWINNERS_SOURCE(4)
      MIN_CONWINNERS_TARGET(4)
      AUTO_ACTIVATE(4);
```

Figure 294. CNOS Statements

Ensure that:

- LOCAL_LU_ALIAS matches the value you specified for the *Alias* field on the Local LU window (Figure 291 on page 223).
- The Network ID and the Partner LU name in the FQ_PARTNER_LU_NAME are only separated by a period.
- PLU_MODE_SESSION_LIMIT matches the value specified for the *Mode session limit* field on the Mode Definition window (Figure 292 on page 224).
- AUTO_ACTIVATE is not greater than PLU_MODE_SESSION_LIMIT (but it must be greater than zero to establish at least one user session).

3. Enter the following command in an OS/2 window:

```
CMVERIFY CSCBCM2
```

This command creates a binary translation of the NDF file which is used to initialize the APPC definitions when Communications Manager is started.

6.3 Change the CICS Client Definitions

The following parameters have to be defined for the CICS Client:

- Matching with parameters of other products:
 - Netname
 - LocalLUName
 - ModeName
- Additional definitions:
 - Server
 - Protocol
 - Driver
 - DriverName

These definitions have to be added to the CICSCLI.INI file which is located in the following directory:

E:\CICSCLI\BIN

where E: is the drive where you installed CICS Client on.

Open the CICSCLI.INI file in edit mode and look for the Server section.

When you installed CICS Client (see 2.2.1, "CICS Client Installation" on page 70) you specified the corresponding values for the CICS for OS/2 server (Server = CICSNETB). Put these definitions in comment by typing a semicolon (;) in the first column of the corresponding lines.

Below the NetBIOS server definition you find sample definitions for a TCP/IP server and an SNA server. Change the SNA server definitions as shown in Figure 295) and save the file.

```
  :
Server = CSCBMVS
  Description = SNA Server OS/2 to CICS/ESA
  Protocol = SNA
  NetName =  USIBMSC . SCMCICSA
  LocalLUName =  STB1255I
  ModeName =  LU62APPB
  :
Driver = SNA
  DriverName = CCLIBMSN
  :
```

Figure 295. CICS Client SNA Server Definition

Notes:

- Ensure that there are no semicolons on the first columns of the Server definition entries as well as the Driver definition entries which are located at the end of the file.
- You have to define a separate server for every server this CICS Client is connecting to. The server name is arbitrary, so you can choose a meaningful one. In our example it is CSCBMVS.

- The Protocol has to match the Driver. The DriverName must be CCLIBMSN.
- The NetName consists of the Network ID and the server APPLID, separated only by a period.

If you want to use more than one server at a time all server definitions in the CICSCLI.INI must be active (without semicolons on the first column). In this case you have to specify in the ECI call of your application program what CICS server has to be called. You can use the Transaction Assistant to do this (see 2.3.3, “How to Use the Transaction Assistant” on page 102).

6.4 VTAM Definitions

The following parameters have to be defined in VTAM of the system where your CICS/ESA is running.

- Matching with parameters of other products:
 - NETID
 - IDBLK/IDNUM
 - LU
 - APPL
 - LOGMODE
- Additional definition:
 - PU

6.4.1 NETID Definition

The NETID of the network where the CICS server is running is defined in the VTAM startup procedure. You can usually find it in SYS1.VTAMLST(ATCSTR00).

The NETID definition looks like:

```

      :
NETID= USIBMSC ,
      :
```

6.4.2 XID/PU/LU Definition

The exchange identifier (XID) as well as the client LU must be defined in VTAM.

An example of this definition is shown in (Figure 296 on page 228):

```

:
STB1255    PU ADDR=01,                                X
            IDBLK= 05D ,IDNUM= B1255 ,                X
            ANS=CONT,DISCNT=NO,                        X
            IRETRY=NO, ISTATUS=ACTIVE,                X
            MAXDATA=265,MAXOUT=1,                    X
            MAXPATH=1,                                X
            PUTYPE=2, SECNET=NO,                      X
            MODETAB=POKMODE, DLOGMOD=DYNRMT,          X
            USSTAB=USSRDYN, LOGAPPL=SCGVAMP,          X
            PACING=2, VPACING=2
*
STB1255A    LU LOCADDR=2
STB1255B    LU LOCADDR=3
STB1255C    LU LOCADDR=4
STB1255D    LU LOCADDR=5
STB1255I    LU LOCADDR=0, DLOGMOD= LU62APPB
:

```

Figure 296. VTAM PU/LU Definition

The PU name (STB1255) is the same name we used for the Local node name in the Communications Manager/2 for OS/2 Warp definition (see Figure 284 on page 219) although it does not necessarily have to be the same.

6.4.3 APPL Definition

Figure 297 shows an example of the VTAM APPL definition for the CICS we are using:

```

SCMACICS  VBUILD TYPE=APPL
*
:
SCMCICSA  APPL  AUTH=(ACQ) ,EAS=1200 ,ACBNAME= SCMCICSA , PARSESS=YES,    X
            MODETAB=SCMODIMS
:

```

Figure 297. APPL Definition

6.4.4 LOGMODE Definition

The MODETAB specified for the APPL (see 6.4.3, “APPL Definition”) must contain the definition of the LOGMODE.

Figure 298 on page 229 shows an example of a LOGMODE definition:

```

SCMODIMS MODETAB
:
LU62APPB  MODEENT LOGMODE=LU62APPB,                                X
          FMPPROF=X'13',TSPPROF=X'07',                                X
          PRIPROT=X'B0',SECPROT=X'B0',                                X
          COMPROT=X'50B5',RUSIZES=X'8585',                            X
          PSERVIC=X'0602000000000000000002F00',                      X
          TYPE=X'00'
:

```

Figure 298. LOGMODE Definition

6.4.5 VTAM Cross-Domain Environment

It could be that your LAN is not in the same network as your server CICS. This means that your workstation is not defined in the same VTAM as your CICS for MVS/ESA. In this case some definitions are different and you need to make additional definitions.

The XID/PU/LU definition must be in the VTAM where your Token Ring is defined and not the server CICS.

In your Communications Manager configuration you have to specify the network ID where your Token Ring is defined in the following fields:

- *C&SM LAN ID* on the Token Ring or Other LAN Types DLC Adapter Parameters window (Figure 283 on page 218)
- *Network ID* on the Local Node Characteristics window (Figure 284 on page 219)

The *Partner network ID* on the Connection to a Host window (Figure 287 on page 221) must contain the cross domain resource manager ID (CDRM). A sample definition of the cross domain resources (CDRSC) is shown in Figure 299.

```

          VBUILD TYPE=CDRSC
          NETWORK NETID=USIBMSC
*
:
STB1255I  CDRSC CDRM=SCG20,                                X
          ISTATUS=ACTIVE
:

```

Figure 299. CDRSC Definition

6.5 CICS for MVS/ESA Definitions

The following parameters have to be defined for CICS for MVS/ESA:

- Matching with parameters of other products:
 - APPLID
 - Netname

- Modename
- Additional definitions:
 - ISC
 - Connection
 - Session
 - Data conversion

Ensure that the DFHISC group is installed for inter system communication. To do this type

```
CEDA I G(DFHISC)
```

on an empty CICS screen.

6.5.1 SIT Definitions

Two parameters have to be defined in the CICS system initialization table (SIT) or as input for the system initialization program (SIP) to allow inter system communication (ISC):

```

:
APPLID= SCMCICSA
ISC=YES
:
```

6.5.2 Connection/Session Definitions

Use the CEDA transaction to define the connections and sessions:

1. Type the following command on an empty CICS screen:

```
CEDA DEF C(C255) G(CSCB)
```

and press Enter.

The Define Connection panel appears (Figure 300 on page 231).

Type the Netname in the *Netname* field and optionally a comment in the *DEscription* field. Press Enter.

```

DEF C(C255) G(CSCB)
OVERTYPE TO MODIFY
CEDA DEFine Connection( C255 )
  Connection      : C255
  Group           : CSCB
  Description      ==> Connection to workstation STB1255I
CONNECTION IDENTIFIERS
  Netname         ==> STB1255I
  INDSys          ==>
REMOTE ATTRIBUTES
  REMOTESYSem     ==>
  REMOTENAME      ==>
  REMOTESYSNet    ==>
CONNECTION PROPERTIES
  ACcessmethod    ==> Vtam      Vtam | IRc | INdirect | Xm
  Protocol        ==> Appc      Appc | Lu61 | Exci
  Conntype        ==>           Generic | Specific
  Singlesess      ==> No        No | Yes
  Datastream      ==> User      User | 3270 | SCs | STRfield | Lms
+ RECORDformat    ==> U         U | Vb

                                SYSID=A    APPLID=SCMCICSA
                                TIME: 23.22.50 DATE: 96.263
DEFINE SUCCESSFUL
PF 1 HELP 2 COM 3 END          6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Figure 300. Define Connection Panel

2. Press F3, type

```
CEDA DEF S(S1255) G(CSCB)
```

and press Enter.

The Define Sessions panel appears (Figure 301 on page 232).

Type C255 in the *Connection* field, the Modname in the *MOdename* field, and optionally a comment in the *DEscription* field. Press Enter.

```

DEF S(S1255) G(CSCB)
OVERTYPE TO MODIFY                                CICS RELEASE = 0410
CEDA DEFINE Sessions( S1255 )
  Sessions    ==> S1255
  Group       ==> CSCB
  Description ==> Session for STB1255I
SESSION IDENTIFIERS
  Connection  ==> C255
  SESSName    ==>
  NETnameq    ==>
  Modename    ==> LU62APPB
SESSION PROPERTIES
  Protocol    ==> Appc           Appc | Lu61 | Exci
  MAXimum     ==> 004 , 000      0-999
  RECEIVEPfx  ==>
  RECEIVECount ==>              1-999
  SENDPfx     ==>
  SENDCount   ==>              1-999
  SENDSize    ==> 04096         1-30720
+ RECEIVESize ==> 04096         1-30720
  S CONNECTION MUST BE SPECIFIED.
                                SYSID=A   APPLID=SCMCICSA

PF 1 HELP 2 COM 3 END          6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Figure 301. Define Sessions Panel

3. Press F3, type

CEDA I G(CSCB)

and press Enter.

The Install panel appears (Figure 274 on page 208), showing the message

INSTALL SUCCESSFUL

6.5.3 Data Conversion Definitions

When sending data in a COMMAREA on an ECI call from the CICS Client to the host CICS the data has to be converted from ASCII to EBCDIC. The server CICS takes care of the conversion but you have to define what has to be converted and how.

Assemble a DFHCNV macro for every resource that has to be converted.

Figure 302 shows an example of the DFHCNV macro definition for the EMPLLU application.

```

DFHCNV TYPE=INITIAL,CLINTCP=437,SRVERCP=037
:
DFHCNV TYPE=ENTRY,RTYPE=PC,RNAME=EMPLLUSV,USREXIT=NO
DFHCNV TYPE=SELECT,OPTION=DEFAULT
DFHCNV TYPE=FIELD,OFFSET=0,DATATYP=CHARACTER,DATALEN=608,      X
      LAST=YES
:

```

Figure 302. DFHCNV Macro Definition

CLINTCP is the code page of the ASCII client.
SRVERCP is the code page of the EBCDIC server.

RTYPE=PC is for a program call which includes an ECI call. The resource to convert is in this case the COMMAREA.

For RNAME you specify the name of the program called by the ECI call.

OFFSET is the starting point within the COMMAREA and DATALEN is the length of the field to be converted.

You can have several TYPE=FIELD statements. The last one must end with LAST=YES.

6.6 Run the Application

You developed and installed the applications in the previous chapters:

- The ECI client (EMPLLU GUI client) in 2.3.2.2, “Installing the Client Part of the Application” on page 95.
- The CICS server program in 5.5, “Prepare the Application on the MVS System” on page 193.

Therefore all you have to do is to run the EMPLUGU program with the SNA server specified in the CICSCLI.INI file. The EMPLUGU.EXE is supposed to be located in the D:\IBMCOBOL\EMPLLU\RT_OS2 directory.

You will get the same result as in 2.3.2.3, “Running the Application” on page 100. But the data you get in the Employee Lookup window is received from DB2 for MVS because the server program you are calling is running on MVS.

Before you run the application read the following tips:

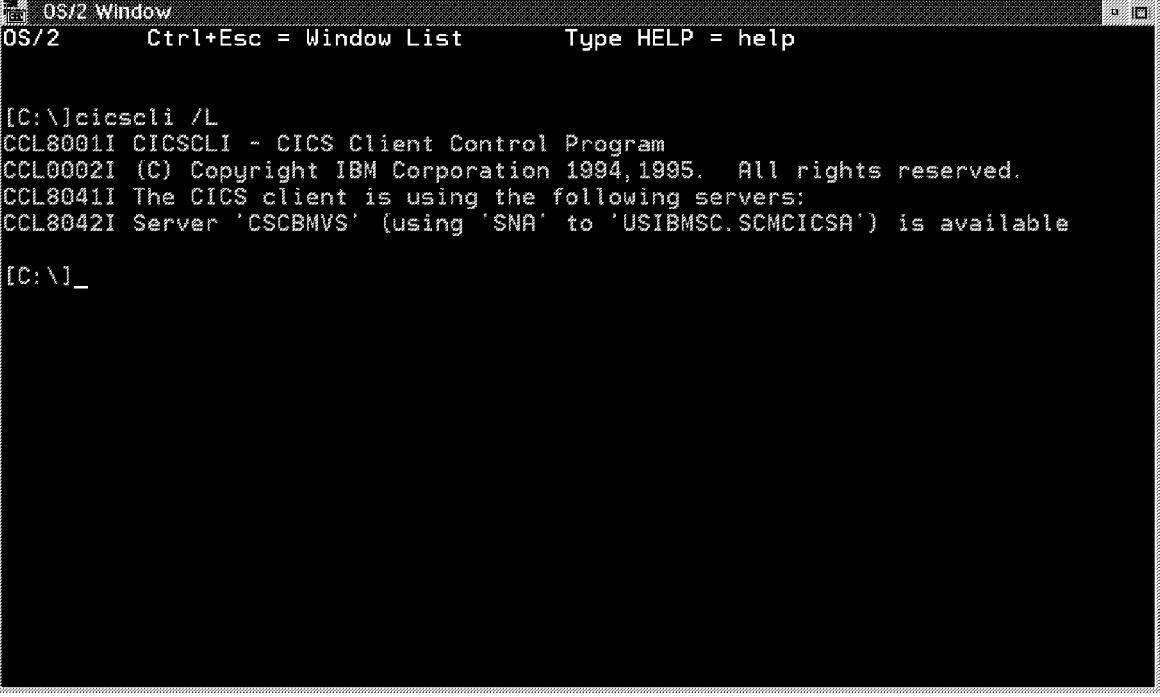
- The ECI call of your GUI client program contains the userid and password parameters. Make sure these parameters contain valid values for your CICS for MVS/ESA environment. If this is not the case either change the values in the client program or define the userid and password in your CICS for MVS/ESA.
- Ensure that your CICS region on MVS is started as well as the CICS-attachment facility for DB2.
- Start Communications Manager/2 for OS/2 Warp on your workstation where the GUI client is running.
- You can test the CICS—CICS connection by issuing the following command in an OS/2 window:

```
CICSCLI /S=CSCBMVS
```

To check the status of your client type

```
CICSCLI /L
```

Figure 303 on page 234 shows the status, telling that the server is available.



```
OS/2 Window
OS/2      Ctrl+Esc = Window List      Type HELP = help

[C:\>cicscli /L
CCL8001I CICSCLI - CICS Client Control Program
CCL0002I (C) Copyright IBM Corporation 1994,1995. All rights reserved.
CCL8041I The CICS client is using the following servers:
CCL8042I Server 'CSCBMVS' (using 'SNA' to 'USIBMSC.SCMCICSA') is available

[C:\>_
```

Figure 303. Client Status OS/2 Window

To stop the client type:

CICSCLI /X=CSCBMVS

- You don't have to start your CICS Client manually. The CICS Client is automatically started when the first ECI call occurs, which is when you click on **Display** on your Employee Lookup window of your GUI program.

Appendix A. Special Notices

This publication is intended to help application developers to install, configure, and use various application development products for client/server applications. The information in this publication is not intended as the specification of any programming interfaces that are provided by IBM VisualAge for COBOL for OS/2, IBM COBOL Set for AIX, or IBM COBOL for MVS and VM. See the PUBLICATIONS section of the IBM Programming Announcement for IBM VisualAge for COBOL for OS/2, IBM COBOL Set for AIX, and IBM COBOL for MVS and VM for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

AIX	AIX/6000
CICS	CICS OS/2
CICS/ESA	CICS/MVS
CICS/6000	DB2
DB2/2	DB2/6000
IBM	MVS/ESA

OS/2
S/390

PS/2
VTAM

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Microsoft, Windows, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

Java and HotJava are trademarks of Sun Microsystems, Inc.

Other trademarks are trademarks of their respective companies.

Appendix B. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

B.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see "How To Get ITSO Redbooks" on page 239.

- *IBM VisualAge for COBOL for OS/2 Primer*, SG24-4605
- *IBM VisualAge for COBOL for OS/2 Workframe User Guide*, SG24-4604
- *IBM VisualAge for COBOL for OS/2 Object Oriented Programming*, SG24-4606

B.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RISC System/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RISC System/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
Application Development Redbooks Collection	SBOF-7290	SK2T-8037
Personal Systems Redbooks Collection	SBOF-7250	SK2T-8042

B.3 Other Publications

These publications are also relevant as further information sources:

- *IBM COBOL for MVS and VM Programming Guide*, SC26-4767
- *IBM COBOL for MVS and VM Language Reference*, SC26-4769
- *IBM COBOL Set for AIX Programming Guide*, SC26-8423
- *IBM VisualAge for COBOL for OS/2 Programming Guide*, SC26-8419

How To Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at URL <http://www.redbooks.ibm.com>.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **PUBORDER** — to order hardcopies in United States
- **GOPHER link to the Internet** - type `GOPHER.WTSCPOK.ITSO.IBM.COM`
- **Tools disks**

To get LIST3820s of redbooks, type one of the following commands:

```
TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
```

To get lists of redbooks:

```
TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET LISTSERV PACKAGE
```

To register for information on workshops, residencies, and redbooks:

```
TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1996
```

For a list of product area specialists in the ITSO:

```
TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
```

- **Redbooks Home Page on the World Wide Web**
<http://w3.itso.ibm.com/redbooks>
- **IBM Direct Publications Catalog on the World Wide Web**
<http://www.elink.ibm.link.ibm.com/pbl/pbl>

IBM employees may obtain LIST3820s of redbooks from this page.

- **REDBOOKS category on INEWS**
- **Online** — send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL
- **Internet Listserver**

With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to announce@webster.ibm.link.ibm.com with the keyword `subscribe` in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** (Do not send credit card information over the Internet) — send orders to:

	IBMMAIL	Internet
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** — send orders to:

IBM Publications Publications Customer Support P.O. Box 29570 Raleigh, NC 27626-0570 USA	IBM Publications 144-4th Avenue, S.W. Calgary, Alberta T2P 3N5 Canada	IBM Direct Services Sortemosevej 21 DK-3450 Allerød Denmark
------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------	----------------------------------------------------------------------

- **Fax** — send orders to:

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1) 415 855 43 29 (Outside USA)** — ask for:

Index # 4421 Abstracts of new redbooks
Index # 4422 IBM redbooks
Index # 4420 Redbooks for last six months

- **Direct Services** - send note to softwareshop@vnet.ibm.com

- **On the World Wide Web**

Redbooks Home Page	http://www.redbooks.ibm.com
IBM Direct Publications Catalog	http://www.elink.ibm.link.ibm.com/pbl/pbl

- **Internet Listserver**

With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to announce@webster.ibm.link.ibm.com with the keyword `subscribe` in the body of the note (leave the subject line blank).

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

First name	Last name
------------	-----------

Company

Address

City	Postal code	Country
------	-------------	---------

Telephone number	Telefax number	VAT number
------------------	----------------	------------

O Invoice to customer number _____

O Credit card number _____

Credit card expiration date	Card issued to	Signature
-----------------------------	----------------	-----------

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

DO NOT SEND CREDIT CARD INFORMATION OVER THE INTERNET.

Glossary

The terms in this glossary are defined in accordance with their meaning in COBOL. These terms may or may not have the same meaning in other languages.

IBM is grateful to the American National Standards Institute (ANSI) for permission to reprint its definitions from the following publications:

- *American National Standard Programming Language COBOL, ANSI X3.23-1985* (Copyright 1985 American National Standards Institute, Inc.), which was prepared by Technical Committee X3J4, which had the task of revising American National Standard COBOL, X3.23-1974.
- *American National Dictionary for Information Processing Systems* (Copyright 1982 by the Computer and Business Equipment Manufacturers Association).

American National Standard definitions are preceded by an asterisk (*).

A

* **abbreviated combined relation condition.** The combined condition that results from the explicit omission of a common subject or a common subject and common relational operator in a consecutive sequence of relation conditions.

abend. Abnormal termination of program.

* **access mode.** The manner in which records are to be operated upon within a file.

* **actual decimal point.** The physical representation, using the decimal point characters period (.) or comma (,), of the decimal point position in a data item.

* **alphabet-name.** A user-defined word, in the SPECIAL-NAMES paragraph of the ENVIRONMENT DIVISION, that assigns a name to a specific character set and/or collating sequence.

* **alphabetic character.** A letter or a space character.

* **alphanumeric character.** Any character in the computer's character set.

alphanumeric-edited character. A character within an alphanumeric character-string that contains at least one B, 0 (zero), or / (slash).

* **alphanumeric function.** A function whose value is composed of a string of one or more characters from the computer's character set.

* **alternate record key.** A key, other than the prime record key, whose contents identify a record within an indexed file.

ANSI (American National Standards Institute). An organization consisting of producers, consumers, and general interest groups, that establishes the procedures by which accredited organizations create and maintain voluntary industry standards in the United States.

* **argument.** An identifier, a literal, an arithmetic expression, or a function-identifier that specifies a value to be used in the evaluation of a function.

* **arithmetic expression.** An identifier of a numeric elementary item, a numeric literal, such identifiers and literals separated by arithmetic operators, two arithmetic expressions separated by an arithmetic operator, or an arithmetic expression enclosed in parentheses.

* **arithmetic operation.** The process caused by the execution of an arithmetic statement, or the evaluation of an arithmetic expression, that results in a mathematically correct solution to the arguments presented.

* **arithmetic operator.** A single character, or a fixed two-character combination that belongs to the following set:

Character	Meaning
+	addition
-	subtraction
*	multiplication
/	division
**	exponentiation

* **arithmetic statement.** A statement that causes an arithmetic operation to be executed. The arithmetic statements are the ADD, COMPUTE, DIVIDE, MULTIPLY, and SUBTRACT statements.

array. In &cel., an aggregate consisting of data objects, each of which may be uniquely referenced by subscripting. Roughly analogous to a COBOL table.

* **ascending key.** A key upon the values of which data is ordered, starting with the lowest value of the key up to the highest value of the key, in accordance with the rules for comparing data items.

ASCII. American National Standard Code for Information Interchange. The standard code, using a coded character set consisting of 7-bit coded characters (8 bits including parity check), used for information interchange between data processing systems, data communication systems, and

associated equipment. The ASCII set consists of control characters and graphic characters.

Note: IBM has defined an extension to ASCII code (characters 128-255).

assignment-name. A name that identifies the organization of a COBOL file and the name by which it is known to the system.

* **assumed decimal point.** A decimal point position that does not involve the existence of an actual character in a data item. The assumed decimal point has logical meaning with no physical representation.

* **AT END condition.** A condition caused:

1. During the execution of a READ statement for a sequentially accessed file, when no next logical record exists in the file, or when the number of significant digits in the relative record number is larger than the size of the relative key data item, or when an optional input file is not present.
2. During the execution of a RETURN statement, when no next logical record exists for the associated sort or merge file.
3. During the execution of a SEARCH statement, when the search operation terminates without satisfying the condition specified in any of the associated WHEN phrases.

B

big-endian. Default format used by the mainframe and the AIX workstation to store binary data. In this format, the least significant digit is on the highest address. Compare with "little-endian."

binary item. A numeric data item represented in binary notation (on the base 2 numbering system). Binary items have a decimal equivalent consisting of the decimal digits 0 through 9, plus an operational sign. The leftmost bit of the item is the operational sign.

binary search. A dichotomizing search in which, at each step of the search, the set of data elements is divided by two; some appropriate action is taken in the case of an odd number.

* **block.** A physical unit of data that is normally composed of one or more logical records. For mass storage files, a block may contain a portion of a logical record. The size of a block has no direct relationship to the size of the file within which the block is contained or to the size of the logical record(s) that are either contained within the block or that overlap the block. The term is synonymous with physical record.

breakpoint. A place in a computer program, usually specified by an instruction, where its execution may

be interrupted by external intervention or by a monitor program.

Btrieve. A key-indexed record management system that allows applications to manage records by key value, sequential access method, or random access method. IBM COBOL supports COBOL sequential and indexed file I-O language through Btrieve.

buffer. A portion of storage used to hold input or output data temporarily.

built-in function. See "intrinsic function".

byte. A string consisting of a certain number of bits, usually eight, treated as a unit, and representing a character.

C

callable services. In &cel., a set of services that can be invoked by a COBOL program using the conventional &cel.-defined call interface, and usable by all programs sharing the &cel. conventions.

called program. A program that is the object of a CALL statement.

* **calling program.** A program that executes a CALL to another program.

case structure. A program processing logic in which a series of conditions is tested in order to make a choice between a number of resulting actions.

cataloged procedure. A set of job control statements placed in a partitioned data set called the procedure library (SYS1.PROCLIB). You can use cataloged procedures to save time and reduce errors coding JCL.

century window. The 100-year interval in which Language Environment assumes all 2-digit years lie. The Language Environment default century window begins 80 years before the system date.

* **character.** The basic indivisible unit of the language.

character position. The amount of physical storage required to store a single standard data format character described as USAGE IS DISPLAY.

character set. All the valid characters for a programming language or a computer system.

* **character-string.** A sequence of contiguous characters that form a COBOL word, a literal, a PICTURE character-string, or a comment-entry. Must be delimited by separators.

checkpoint. A point at which information about the status of a job and the system can be recorded so that the job step can be later restarted.

* **class.** The entity that defines common behavior and implementation for zero, one, or more objects. The objects that share the same implementation are considered to be objects of the same class.

* **class condition.** The proposition, for which a truth value can be determined, that the content of an item is wholly alphabetic, is wholly numeric, or consists exclusively of those characters listed in the definition of a class-name.

* **Class Definition.** The COBOL source unit that defines a class.

* **class identification entry.** An entry in the CLASS-ID paragraph of the IDENTIFICATION DIVISION which contains clauses that specify the class-name and assign selected attributes to the class definition.

* **class-name.** A user-defined word defined in the SPECIAL-NAMES paragraph of the ENVIRONMENT DIVISION that assigns a name to the proposition for which a truth value can be defined, that the content of a data item consists exclusively of those characters listed in the definition of the class-name.

class object. The run-time object representing a SOM class.

* **clause.** An ordered set of consecutive COBOL character-strings whose purpose is to specify an attribute of an entry.

CMS (Conversational Monitor System). A virtual machine operating system that provides general interactive, time-sharing, problem solving, and program development capabilities, and that operates only under the control of the VM/SP control program.

* **COBOL character set.** The complete COBOL character set consists of the characters listed below:

Character	Meaning
0,1,...,9	digit
A,B,...,Z	uppercase letter
a,b,...,z	lowercase letter
	space
+	plus sign
-	minus sign (hyphen)
*	asterisk
/	slant (virgule, slash)
=	equal sign
\$	currency sign
,	comma (decimal point)
;	semicolon
.	period (decimal point, full stop)
"	quotation mark
(left parenthesis

)	right parenthesis
>	greater than symbol
<	less than symbol
:	colon

* **COBOL word.** See "word."

code page. An assignment of graphic characters and control function meanings to all code points; for example, assignment of characters and meanings to 256 code points for 8-bit code, assignment of characters and meanings to 128 code points for 7-bit code.

* **collating sequence.** The sequence in which the characters that are acceptable to a computer are ordered for purposes of sorting, merging, comparing, and for processing indexed files sequentially.

* **column.** A character position within a print line. The columns are numbered from 1, by 1, starting at the leftmost character position of the print line and extending to the rightmost position of the print line.

* **combined condition.** A condition that is the result of connecting two or more conditions with the AND or the OR logical operator.

* **comment-entry.** An entry in the IDENTIFICATION DIVISION that may be any combination of characters from the computer's character set.

* **comment line.** A source program line represented by an asterisk (*) in the indicator area of the line and any characters from the computer's character set in area A and area B of that line. The comment line serves only for documentation in a program. A special form of comment line represented by a slant (/) in the indicator area of the line and any characters from the computer's character set in area A and area B of that line causes page ejection prior to printing the comment.

* **common program.** A program which, despite being directly contained within another program, may be called from any program directly or indirectly contained in that other program.

* **compile.** (1) To translate a program expressed in a high-level language into a program expressed in an intermediate language, assembly language, or a computer language. (2) To prepare a machine language program from a computer program written in another programming language by making use of the overall logic structure of the program, or generating more than one computer instruction for each symbolic statement, or both, as well as performing the function of an assembler.

* **compile time.** The time at which a COBOL source program is translated, by a COBOL compiler, to a COBOL object program.

compiler. A program that translates a program written in a higher level language into a machine language object program.

compiler directing statement. A statement, beginning with a compiler directing verb, that causes the compiler to take a specific action during compilation.

compiler directing statement. A statement that specifies actions to be taken by the compiler during processing of a COBOL source program. Compiler directives are contained in the COBOL source program. Thus, you can specify different suboptions of the directive within the source program by using multiple compiler directive statements in the program.

* **complex condition.** A condition in which one or more logical operators act upon one or more conditions. (See also “negated simple condition,” “combined condition,” and “negated combined condition.”)

* **computer-name.** A system-name that identifies the computer upon which the program is to be compiled or run.

condition. An exception that has been enabled, or recognized, by &cel. and thus is eligible to activate user and language condition handlers. Any alteration to the normal programmed flow of an application. Conditions can be detected by the hardware/operating system and results in an interrupt. They can also be detected by language-specific generated code or language library code.

* **condition.** A status of a program at run time for which a truth value can be determined. Where the term ‘condition’ (condition-1, condition-2,...) appears in these language specifications in or in reference to ‘condition’ (condition-1, condition-2,...) of a general format, it is a conditional expression consisting of either a simple condition optionally parenthesized, or a combined condition consisting of the syntactically correct combination of simple conditions, logical operators, and parentheses, for which a truth value can be determined.

* **conditional expression.** A simple condition or a complex condition specified in an EVALUATE, IF, PERFORM, or SEARCH statement. (See also “simple condition” and “complex condition.”)

* **conditional phrase.** A conditional phrase specifies the action to be taken upon determination of the truth value of a condition resulting from the execution of a conditional statement.

* **conditional statement.** A statement specifying that the truth value of a condition is to be determined and that the subsequent action of the object program is dependent on this truth value.

* **conditional variable.** A data item one or more values of which has a condition-name assigned to it.

* **condition-name.** A user-defined word that assigns a name to a subset of values that a conditional variable may assume; or a user-defined word assigned to a status of an implementor defined switch or device. When ‘condition-name’ is used in the general formats, it represents a unique data item reference consisting of a syntactically correct combination of a ‘condition-name’, together with qualifiers and subscripts, as required for uniqueness of reference.

* **condition-name condition.** The proposition, for which a truth value can be determined, that the value of a conditional variable is a member of the set of values attributed to a condition-name associated with the conditional variable.

* **CONFIGURATION SECTION.** A section of the ENVIRONMENT DIVISION that describes overall specifications of source and object programs and class definitions.

CONSOLE. A COBOL environment-name associated with the operator console.

* **contiguous items.** Items that are described by consecutive entries in the Data Division, and that bear a definite hierarchic relationship to each other.

CORBA. The Common Object Request Broker Architecture established by the Object Management Group. IBM's *Interface Definition Language* used to describe the *interface* for SOM classes is fully compliant with CORBA standards.

* **counter.** A data item used for storing numbers or number representations in a manner that permits these numbers to be increased or decreased by the value of another number, or to be changed or reset to zero or to an arbitrary positive or negative value.

cross-reference listing. The portion of the compiler listing that contains information on where files, fields, and indicators are defined, referenced, and modified in a program.

* **currency sign.** The character ‘\$’ of the COBOL character set or that character defined by the CURRENCY compiler option. If the NOCURRENCY compiler option is in effect, the currency sign is defined as the character ‘\$’.

currency symbol. The character defined by the CURRENCY compiler option or by the CURRENCY SIGN clause in the SPECIAL-NAMES paragraph. If the NOCURRENCY compiler option is in effect for a COBOL source program and the CURRENCY SIGN clause is also **not** present in the source program, the currency symbol is identical to the currency sign.

* **current record.** In file processing, the record that is available in the record area associated with a file.

* **current volume pointer.** A conceptual entity that points to the current volume of a sequential file.

D

* **data clause.** A clause, appearing in a data description entry in the DATA DIVISION of a COBOL program, that provides information describing a particular attribute of a data item.

* **data description entry .** An entry in the DATA DIVISION of a COBOL program that is composed of a level-number followed by a data-name, if required, and then followed by a set of data clauses, as required.

DATA DIVISION. One of the four main components of a COBOL program, class definition, or method definition. The DATA DIVISION describes the data to be processed by the object program, class, or method: files to be used and the records contained within them; internal working-storage records that will be needed; data to be made available in more than one program in the COBOL run unit. (Note, the Class DATA DIVISION contains only the WORKING-STORAGE SECTION.)

* **data item.** A unit of data (excluding literals) defined by a COBOL program or by the rules for function evaluation.

* **data-name.** A user-defined word that names a data item described in a data description entry. When used in the general formats, 'data-name' represents a word that must not be reference-modified, subscripted or qualified unless specifically permitted by the rules for the format.

DBCS (Double-Byte Character Set). See "Double-Byte Character Set (DBCS)."

* **debugging line.** A debugging line is any line with a 'D' in the indicator area of the line.

* **debugging section.** A section that contains a USE FOR DEBUGGING statement.

* **declarative sentence.** A compiler directing sentence consisting of a single USE statement terminated by the separator period.

* **declaratives.** A set of one or more special purpose sections, written at the beginning of the Procedure Division, the first of which is preceded by the key word DECLARATIVES and the last of which is followed by the key words END DECLARATIVES. A declarative is composed of a section header, followed by a USE compiler directing sentence, followed by a set of zero, one, or more associated paragraphs.

* **de-edit.** The logical removal of all editing characters from a numeric edited data item in order to determine that item's unedited numeric value.

* **delimited scope statement.** Any statement that includes its explicit scope terminator.

* **delimiter.** A character or a sequence of contiguous characters that identify the end of a string of characters and separate that string of characters from the following string of characters. A delimiter is not part of the string of characters that it delimits.

* **descending key.** A key upon the values of which data is ordered starting with the highest value of key down to the lowest value of key, in accordance with the rules for comparing data items.

digit. Any of the numerals from 0 through 9. In COBOL, the term is not used in reference to any other symbol.

* **digit position.** The amount of physical storage required to store a single digit. This amount may vary depending on the usage specified in the data description entry that defines the data item.

* **direct access.** The facility to obtain data from storage devices or to enter data into a storage device in such a way that the process depends only on the location of that data and not on a reference to data previously accessed.

* **division.** A collection of zero, one or more sections or paragraphs, called the division body, that are formed and combined in accordance with a specific set of rules. Each division consists of the division header and the related division body. There are four (4) divisions in a COBOL program: Identification, Environment, Data, and Procedure.

* **division header.** A combination of words followed by a separator period that indicates the beginning of a division. The division headers are:

IDENTIFICATION DIVISION.
ENVIRONMENT DIVISION.
DATA DIVISION.
PROCEDURE DIVISION.

do construction. In structured programming, a DO statement is used to group a number of statements in a procedure. In COBOL, an in-line PERFORM statement functions in the same way.

do-until. In structured programming, a do-until loop will be executed at least once, and until a given condition is true. In COBOL, a TEST AFTER phrase used with the PERFORM statement functions in the same way.

do-while. In structured programming, a do-while loop will be executed if, and while, a given condition is

true. In COBOL, a TEST BEFORE phrase used with the PERFORM statement functions in the same way.

Double-Byte Character Set (DBCS). A set of characters in which each character is represented by two bytes. Languages such as Japanese, Chinese, and Korean, which contain more symbols than can be represented by 256 code points, require Double-Byte Character Sets. Because each character requires two bytes, entering, displaying, and printing DBCS characters requires hardware and supporting software that are DBCS-capable.

*** dynamic access.** An access mode in which specific logical records can be obtained from or placed into a mass storage file in a nonsequential manner and obtained from a file in a sequential manner during the scope of the same OPEN statement.

Dynamic Storage Area (DSA). Dynamically acquired storage composed of a register save area and an area available for dynamic storage allocation (such as program variables). DSAs are generally allocated within STACK segments managed by &cel..

E

*** EBCDIC (Extended Binary-Coded Decimal Interchange Code).** A coded character set consisting of 8-bit coded characters.

EBCDIC character. Any one of the symbols included in the 8-bit EBCDIC (Extended Binary-Coded-Decimal Interchange Code) set.

edited data item. A data item that has been modified by suppressing zeroes and/or inserting editing characters.

*** editing character.** A single character or a fixed two-character combination belonging to the following set:

Character	Meaning
	space
0	zero
+	plus
-	minus
CR	credit
DB	debit
Z	zero suppress
*	check protect
\$	currency sign
,	comma (decimal point)
.	period (decimal point)
/	slant (virgule, slash)

element (text element). One logical unit of a string of text, such as the description of a single data item or verb, preceded by a unique code identifying the element type.

*** elementary item.** A data item that is described as not being further logically subdivided.

enclave. When running under the &cel. product, an enclave is analogous to a run unit. An enclave can create other enclaves on MVS and CMS by a LINK, on CMS by CMSCALL, and the use of the system () function of C.

***end class header.** A combination of words, followed by a separator period, that indicates the end of a COBOL class definition. The end class header is:

END CLASS class-name.

***end method header.** A combination of words, followed by a separator period, that indicates the end of a COBOL method definition. The end method header is:

END METHOD method-name.

*** end of Procedure Division.** The physical position of a COBOL source program after which no further procedures appear.

*** end program header.** A combination of words, followed by a separator period, that indicates the end of a COBOL source program. The end program header is:

END PROGRAM program-name.

*** entry.** Any descriptive set of consecutive clauses terminated by a separator period and written in the IDENTIFICATION DIVISION, ENVIRONMENT DIVISION, or DATA DIVISION of a COBOL program.

*** environment clause.** A clause that appears as part of an ENVIRONMENT DIVISION entry.

ENVIRONMENT DIVISION. One of the four main component parts of a COBOL program, class definition, or method definition. The ENVIRONMENT DIVISION describes the computers upon which the source program is compiled and those on which the object program is executed, and provides a linkage between the logical concept of files and their records, and the physical aspects of the devices on which files are stored.

environment-name. A name, specified by IBM, that identifies system logical units, printer and card punch control characters, report codes, and/or program switches. When an environment-name is associated with a mnemonic-name in the ENVIRONMENT DIVISION, the mnemonic-name may then be substituted in any format in which such substitution is valid.

environment variable. Any of a number of variables that describe the way an operating system is going to run and the devices it is going to recognize.

execution time. See "run time."

execution-time environment. See “run-time environment.”

* **explicit scope terminator.** A reserved word that terminates the scope of a particular Procedure Division statement.

exponent. A number, indicating the power to which another number (the base) is to be raised. Positive exponents denote multiplication, negative exponents denote division, fractional exponents denote a root of a quantity. In COBOL, an exponential expression is indicated with the symbol “**” followed by the exponent.

* **expression.** An arithmetic or conditional expression.

* **extend mode.** The state of a file after execution of an OPEN statement, with the EXTEND phrase specified for that file, and before the execution of a CLOSE statement, without the REEL or UNIT phrase for that file.

extensions. Certain COBOL syntax and semantics supported by IBM compilers in addition to those described in ANSI Standard.

* **external data.** The data described in a program as external data items and external file connectors.

* **external data item.** A data item which is described as part of an external record in one or more programs of a run unit and which itself may be referenced from any program in which it is described.

* **external data record.** A logical record which is described in one or more programs of a run unit and whose constituent data items may be referenced from any program in which they are described.

external decimal item. A format for representing numbers in which the digit is contained in bits 4 through 7 and the sign is contained in bits 0 through 3 of the rightmost byte. Bits 0 through 3 of all other bytes contain 1’s (hex F). For example, the decimal value of +123 is represented as 1111 0001 1111 0010 1111 0011. (Also known as “zoned decimal item.”)

* **external file connector.** A file connector which is accessible to one or more object programs in the run unit.

external floating-point item. A format for representing numbers in which a real number is represented by a pair of distinct numerals. In a floating-point representation, the real number is the product of the fixed-point part (the first numeral), and a value obtained by raising the implicit floating-point base to a power denoted by the exponent (the second numeral).

For example, a floating-point representation of the number 0.0001234 is: 0.1234 -3, where 0.1234 is the mantissa and -3 is the exponent.

* **external switch.** A hardware or software device, defined and named by the implementor, which is used to indicate that one of two alternate states exists.

F

* **figurative constant.** A compiler-generated value referenced through the use of certain reserved words.

* **file.** A collection of logical records.

* **file attribute conflict condition.** An unsuccessful attempt has been made to execute an input-output operation on a file and the file attributes, as specified for that file in the program, do not match the fixed attributes for that file.

* **file clause.** A clause that appears as part of any of the following DATA DIVISION entries: file description entry (FD entry) and sort-merge file description entry (SD entry).

* **file connector.** A storage area which contains information about a file and is used as the linkage between a file-name and a physical file and between a file-name and its associated record area.

File-Control. The name of an ENVIRONMENT DIVISION paragraph in which the data files for a given source program are declared.

* **file control entry.** A SELECT clause and all its subordinate clauses which declare the relevant physical attributes of a file.

* **file description entry.** An entry in the File Section of the DATA DIVISION that is composed of the level indicator FD, followed by a file-name, and then followed by a set of file clauses as required.

* **file-name.** A user-defined word that names a file connector described in a file description entry or a sort-merge file description entry within the File Section of the DATA DIVISION.

* **file organization.** The permanent logical file structure established at the time that a file is created.

* **file position indicator.** A conceptual entity that contains the value of the current key within the key of reference for an indexed file, or the record number of the current record for a sequential file, or the relative record number of the current record for a relative file, or indicates that no next logical record exists, or that an optional input file is not present, or that the at end condition already exists, or that no valid next record has been established.

* **File Section.** The section of the DATA DIVISION that contains file description entries and sort-merge file description entries together with their associated record descriptions.

file system. The collection of files and file management structures on a physical or logical mass storage device, such as a diskette or minidisk.

* **fixed file attributes.** Information about a file which is established when a file is created and cannot subsequently be changed during the existence of the file. These attributes include the organization of the file (sequential, relative, or indexed), the prime record key, the alternate record keys, the code set, the minimum and maximum record size, the record type (fixed or variable), the collating sequence of the keys for indexed files, the blocking factor, the padding character, and the record delimiter.

* **fixed length record.** A record associated with a file whose file description or sort-merge description entry requires that all records contain the same number of character positions.

fixed-point number. A numeric data item defined with a PICTURE clause that specifies the location of an optional sign, the number of digits it contains, and the location of an optional decimal point. The format may be either binary, packed decimal, or external decimal.

floating-point number. A numeric data item containing a fraction and an exponent. Its value is obtained by multiplying the fraction by the base of the numeric data item raised to the power specified by the exponent.

* **format.** A specific arrangement of a set of data.

* **function.** A temporary data item whose value is determined at the time the function is referenced during the execution of a statement.

* **function-identifier.** A syntactically correct combination of character-strings and separators that references a function. The data item represented by a function is uniquely identified by a function-name with its arguments, if any. A function-identifier may include a reference-modifier. A function-identifier that references an alphanumeric function may be specified anywhere in the general formats that an identifier may be specified, subject to certain restrictions. A function-identifier that references an integer or numeric function may be referenced anywhere in the general formats that an arithmetic expression may be specified.

function-name. A word that names the mechanism whose invocation, along with required arguments, determines the value of a function.

G

* **global name.** A name which is declared in only one program but which may be referenced from that program and from any program contained within that program. Condition-names, data-names, file-names, record-names, report-names, and some special registers may be global names.

* **group item.** A data item that is composed of subordinate data items.

H

header label. (1) A file label or data set label that precedes the data records on a unit of recording media. (2) Synonym for beginning-of-file label.

* **high order end.** The leftmost character of a string of characters.

I

IBM COBOL extension. Certain COBOL syntax and semantics supported by IBM compilers in addition to those described in ANSI Standard.

IDENTIFICATION DIVISION. One of the four main component parts of a COBOL program, class definition, or method definition. The IDENTIFICATION DIVISION identifies the program name, class name, or method name. The IDENTIFICATION DIVISION may include the following documentation: author name, installation, or date.

* **identifier.** A syntactically correct combination of character-strings and separators that names a data item. When referencing a data item that is not a function, an identifier consists of a data-name, together with its qualifiers, subscripts, and reference-modifier, as required for uniqueness of reference. When referencing a data item which is a function, a function-identifier is used.

IGZCBSN. The &cobol. bootstrap routine. It must be link-edited with any module that contains a &cobol. program.

* **imperative statement.** A statement that either begins with an imperative verb and specifies an unconditional action to be taken or is a conditional statement that is delimited by its explicit scope terminator (delimited scope statement). An imperative statement may consist of a sequence of imperative statements.

* **implicit scope terminator.** A separator period which terminates the scope of any preceding unterminated statement, or a phrase of a statement which by its occurrence indicates the end of the scope of any statement contained within the preceding phrase.

* **index.** A computer storage area or register, the content of which represents the identification of a particular element in a table.

* **index data item.** A data item in which the values associated with an index-name can be stored in a form specified by the implementor.

indexed data-name. An identifier that is composed of a data-name, followed by one or more index-names enclosed in parentheses.

* **indexed file.** A file with indexed organization.

* **indexed organization.** The permanent logical file structure in which each record is identified by the value of one or more keys within that record.

indexing. Synonymous with subscripting using index-names.

* **index-name.** A user-defined word that names an index associated with a specific table.

* **inheritance (for classes).** A mechanism for using the implementation of one or more *classes* as the basis for another class. A *sub-class* inherits from one or more *super-classes*. By definition the inheriting class conforms to the inherited classes.

* **initial program.** A program that is placed into an initial state every time the program is called in a run unit.

* **initial state.** The state of a program when it is first called in a run unit.

inline. In a program, instructions that are executed sequentially, without branching to routines, subroutines, or other programs.

* **input file.** A file that is opened in the INPUT mode.

* **input mode.** The state of a file after execution of an OPEN statement, with the INPUT phrase specified, for that file and before the execution of a CLOSE statement, without the REEL or UNIT phrase for that file.

* **input-output file.** A file that is opened in the I-O mode.

* **INPUT-OUTPUT SECTION.** The section of the ENVIRONMENT DIVISION that names the files and the external media required by an object program or method and that provides information required for transmission and handling of data during execution of the object program or method definition.

* **Input-Output statement.** A statement that causes files to be processed by performing operations upon individual records or upon the file as a unit. The input-output statements are: ACCEPT (with the

identifier phrase), CLOSE, DELETE, DISPLAY, OPEN, READ, REWRITE, SET (with the TO ON or TO OFF phrase), START, and WRITE.

* **input procedure.** A set of statements, to which control is given during the execution of a SORT statement, for the purpose of controlling the release of specified records to be sorted.

instance data. Data defining the state of an object. The instance data introduced by a class is defined in the WORKING-STORAGE SECTION of the DATA DIVISION of the class definition. The state of an object also includes the state of the instance variables introduced by base classes that are inherited by the current class. A separate copy of the instance data is created for each object instance.

* **integer.** (1) A numeric literal that does not include any digit positions to the right of the decimal point.

(2) A numeric data item defined in the DATA DIVISION that does not include any digit positions to the right of the decimal point.

(3) A numeric function whose definition provides that all digits to the right of the decimal point are zero in the returned value for any possible evaluation of the function.

integer function. A function whose category is numeric and whose definition does not include any digit positions to the right of the decimal point.

interface. The information that a *client* must know to use a *class*—the names of its *attributes* and the signatures of its *methods*. With direct-to-SOM compilers such as COBOL, the interface to a class may be defined by native language syntax for class definitions. Classes implemented in other languages might have their interfaces defined directly in SOM Interface Definition Language (IDL). The COBOL compiler has a compiler option, IDLGEN, to automatically generate IDL for a COBOL class.

Interface Definition Language (IDL). The formal language (independent of any programming language) by which the *interface* for a class of *objects* is defined in a IDL file, which the SOM compiler then interprets to create an implementation template file and binding files. SOM's Interface Definition Language is fully compliant with standards established by the Object Management Group's Common Object Request Broker Architecture (CORBA).

interlanguage communication (ILC). The ability of routines written in different programming languages to communicate. ILC support allows the application writer to readily build applications from component routines written in a variety of languages.

intermediate result. An intermediate field containing the results of a succession of arithmetic operations.

* **internal data.** The data described in a program excluding all external data items and external file connectors. Items described in the LINKAGE SECTION of a program are treated as internal data.

* **internal data item.** A data item which is described in one program in a run unit. An internal data item may have a global name.

internal decimal item. A format in which each byte in a field except the rightmost byte represents two numeric digits. The rightmost byte contains one digit and the sign. For example, the decimal value +123 is represented as 0001 0010 0011 1111. (Also known as packed decimal.)

* **internal file connector.** A file connector which is accessible to only one object program in the run unit.

* **intra-record data structure.** The entire collection of groups and elementary data items from a logical record which is defined by a contiguous subset of the data description entries which describe that record. These data description entries include all entries whose level-number is greater than the level-number of the first data description entry describing the intra-record data structure.

intrinsic function. A pre-defined function, such as a commonly used arithmetic function, called by a built-in function reference.

* **invalid key condition.** A condition, at object time, caused when a specific value of the key associated with an indexed or relative file is determined to be invalid.

* **I-O-CONTROL.** The name of an ENVIRONMENT DIVISION paragraph in which object program requirements for rerun points, sharing of same areas by several data files, and multiple file storage on a single input-output device are specified.

* **I-O-CONTROL entry.** An entry in the I-O-CONTROL paragraph of the ENVIRONMENT DIVISION which contains clauses that provide information required for the transmission and handling of data on named files during the execution of a program.

* **I-O-Mode.** The state of a file after execution of an OPEN statement, with the I-O phrase specified, for that file and before the execution of a CLOSE statement without the REEL or UNIT phase for that file.

* **I-O status.** A conceptual entity which contains the two-character value indicating the resulting status of an input-output operation. This value is made available to the program through the use of the FILE STATUS clause in the file control entry for the file.

iteration structure. A program processing logic in which a series of statements is repeated while a condition is true or until a condition is true.

K

K. When referring to storage capacity, two to the tenth power; 1024 in decimal notation.

* **key.** A data item that identifies the location of a record, or a set of data items which serve to identify the ordering of data.

* **key of reference.** The key, either prime or alternate, currently being used to access records within an indexed file.

* **key word.** A reserved word or function-name whose presence is required when the format in which the word appears is used in a source program.

kilobyte (KB). One kilobyte equals 1024 bytes.

L

* **language-name.** A system-name that specifies a particular programming language.

Language Environment-conforming. A characteristic of compiler products (&cobol370., &cobol., AD/Cycle C/370, C/C++ for MVS and VM, PL/I for MVS and VM) that produce object code conforming to the Language Environment format.

last-used state. A program is in last-used state if its internal values remain the same as when the program was exited (are not reset to their initial values).

* **letter.** A character belonging to one of the following two sets:

1. Uppercase letters: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z
2. Lowercase letters: a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z

* **level indicator.** Two alphabetic characters that identify a specific type of file or a position in a hierarchy. The level indicators in the DATA DIVISION are: CD, FD, and SD.

* **level-number.** A user-defined word, expressed as a two digit number, which indicates the hierarchical position of a data item or the special properties of a data description entry. Level-numbers in the range from 1 through 49 indicate the position of a data item in the hierarchical structure of a logical record. Level-numbers in the range 1 through 9 may be written either as a single digit or as a zero followed by a significant digit. Level-numbers 66, 77 and 88 identify special properties of a data description entry.

* **library-name.** A user-defined word that names a COBOL library that is to be used by the compiler for a given source program compilation.

* **library text.** A sequence of text words, comment lines, the separator space, or the separator pseudo-text delimiter in a COBOL library.

LILIAN DATE. The number of days since the beginning of the Gregorian calendar. Day one is Friday, October 15, 1582. The Lilian date format is named in honor of Luigi Lilio, the creator of the Gregorian calendar.

* **LINAGE-COUNTER.** A special register whose value points to the current position within the page body.

LINKAGE SECTION. The section in the DATA DIVISION of the called program that describes data items available from the calling program. These data items may be referred to by both the calling and called program.

literal. A character-string whose value is specified either by the ordered set of characters comprising the string, or by the use of a figurative constant.

local. A set of attributes for a program execution environment indicating culturally sensitive considerations, such as: character code page, collating sequence, date/time format, monetary value representation, numeric value representation, or language.

* **LOCAL-STORAGE SECTION.** The section of the DATA DIVISION that defines storage that is allocated and freed on a per-invocation basis, depending on the value assigned in their VALUE clauses.

* **logical operator.** One of the reserved words AND, OR, or NOT. In the formation of a condition, either AND, or OR, or both can be used as logical connectives. NOT can be used for logical negation.

* **logical record.** The most inclusive data item. The level-number for a record is 01. A record may be either an elementary item or a group of items. The term is synonymous with record.

* **low order end.** The rightmost character of a string of characters.

M

main program. In a hierarchy of programs and subroutines, the first program to receive control when the programs are run.

* **mass storage.** A storage medium in which data may be organized and maintained in both a sequential and nonsequential manner.

* **mass storage device.** A device having a large storage capacity; for example, magnetic disk, magnetic drum.

* **mass storage file.** A collection of records that is assigned to a mass storage medium.

* **megabyte (M).** One megabyte equals 1,048,576 bytes.

* **merge file.** A collection of records to be merged by a MERGE statement. The merge file is created and can be used only by the merge function.

metaclass. A SOM class whose instances are SOM class-objects. The methods defined in metaclasses are executed without requiring any object instances of the class to exist, and are frequently used to create instances of the class.

method. Procedural code that defines one of the operations supported by an object, and that is executed by an INVOKE statement on that object.

* **Method Definition.** The COBOL source unit that defines a method.

* **method identification entry.** An entry in the METHOD-ID paragraph of the IDENTIFICATION DIVISION which contains clauses that specify the method-name and assign selected attributes to the method definition.

* **method-name.** A user-defined word that identifies a method.

* **mnemonic-name.** A user-defined word that is associated in the ENVIRONMENT DIVISION with a specified implementor-name.

multitasking. Mode of operation that provides for the concurrent, or interleaved, execution of two or more tasks. When running under the &cel. product, multitasking is synonymous with *multithreading*.

MVS/XA* (Multiple Virtual Storage/Extended Architecture). An IBM operating system that manages multiple virtual address spaces in IBM processors operating in extended architecture mode. MVS/XA supports the 31-bit addressing mechanism of extended architecture mode, and therefore, allows an address space as large as 2³¹ bytes (2048 megabytes or 2 gigabytes).

N

name. A word composed of not more than 30 characters that defines a COBOL operand.

* **native character set.** The implementor-defined character set associated with the computer specified in the OBJECT-COMPUTER paragraph.

* **native collating sequence.** The implementor-defined collating sequence associated with the computer specified in the OBJECT-COMPUTER paragraph.

* **negated combined condition.** The 'NOT' logical operator immediately followed by a parenthesized combined condition.

* **negated simple condition.** The 'NOT' logical operator immediately followed by a simple condition.

nested program. A program that is directly contained within another program.

* **next executable sentence.** The next sentence to which control will be transferred after execution of the current statement is complete.

* **next executable statement.** The next statement to which control will be transferred after execution of the current statement is complete.

* **next record.** The record that logically follows the current record of a file.

* **noncontiguous items.** Elementary data items in the WORKING-STORAGE and LINKAGE SECTIONS that bear no hierarchic relationship to other data items.

* **nonnumeric item.** A data item whose description permits its content to be composed of any combination of characters taken from the computer's character set. Certain categories of nonnumeric items may be formed from more restricted character sets.

* **nonnumeric literal.** A literal bounded by quotation marks. The string of characters may include any character in the computer's character set.

null. Figurative constant used to assign the value of an invalid address to pointer data items. NULLS can be used wherever NULL can be used.

* **numeric character.** A character that belongs to the following set of digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

numeric-edited item. A numeric item that is in such a form that it may be used in printed output. It may consist of external decimal digits from 0 through 9, the decimal point, commas, the dollar sign, editing sign control symbols, plus other editing symbols.

* **numeric function.** A function whose class and category are numeric but which for some possible evaluation does not satisfy the requirements of integer functions.

* **numeric item.** A data item whose description restricts its content to a value represented by characters chosen from the digits from '0' through '9'; if signed, the item may also contain a '+', '-', or other representation of an operational sign.

* **numeric literal.** A literal composed of one or more numeric characters that may contain either a decimal point, or an algebraic sign, or both. The decimal point must not be the rightmost character. The algebraic sign, if present, must be the leftmost character.

O

object. An entity that has state (its data values) and operations (its methods). An object is a way to encapsulate state and behavior.

object code. Output from a compiler or assembler that is itself executable machine code or is suitable for processing to produce executable machine code.

* **OBJECT-COMPUTER.** The name of an ENVIRONMENT DIVISION paragraph in which the computer environment, within which the object program is executed, is described.

* **object computer entry.** An entry in the OBJECT-COMPUTER paragraph of the ENVIRONMENT DIVISION which contains clauses that describe the computer environment in which the object program is to be executed.

object deck. A portion of an object program suitable as input to a linkage editor. Synonymous with *object module* and *text deck*.

object module. Synonym for *object deck* or *text deck*.

* **object of entry.** A set of operands and reserved words, within a DATA DIVISION entry of a COBOL program, that immediately follows the subject of the entry.

* **object program.** A set or group of executable machine language instructions and other material designed to interact with data to provide problem solutions. In this context, an object program is generally the machine language result of the operation of a COBOL compiler on a source program. Where there is no danger of ambiguity, the word 'program' alone may be used in place of the phrase 'object program.'

* **object time.** The time at which an object program is executed. The term is synonymous with execution time.

* **obsolete element.** A COBOL language element in Standard COBOL that is to be deleted from the next revision of Standard COBOL.

ODO object. In the example below,

```
WORKING-STORAGE SECTION
01  TABLE-1.
    05  X                                PICS9.
    05  Y OCCURS 3 TIMES
        DEPENDING ON X                PIC X.
```


X is the object of the OCCURS DEPENDING ON clause (ODO object). The value of the ODO object determines how many of the ODO subject appear in the table.

ODO subject. In the example above, Y is the subject of the OCCURS DEPENDING ON clause (ODO subject). The number of Y ODO subjects that appear in the table depends on the value of X.

* **open mode.** The state of a file after execution of an OPEN statement for that file and before the execution of a CLOSE statement without the REEL or UNIT phrase for that file. The particular open mode is specified in the OPEN statement as either INPUT, OUTPUT, I-O or EXTEND.

* **operand.** Whereas the general definition of operand is "that component which is operated upon," for the purposes of this document, any lowercase word (or words) that appears in a statement or entry format may be considered to be an operand and, as such, is an implied reference to the data indicated by the operand.

* **operational sign.** An algebraic sign, associated with a numeric data item or a numeric literal, to indicate whether its value is positive or negative.

* **optional file.** A file which is declared as being not necessarily present each time the object program is executed. The object program causes an interrogation for the presence or absence of the file.

* **optional word.** A reserved word that is included in a specific format only to improve the readability of the language and whose presence is optional to the user when the format in which the word appears is used in a source program.

OS/2 (Operating System/2*). A multi-tasking operating system for the IBM Personal Computer family that allows you to run both DOS mode and OS/2 mode programs.

* **output file.** A file that is opened in either the OUTPUT mode or EXTEND mode.

* **output mode.** The state of a file after execution of an OPEN statement, with the OUTPUT or EXTEND phrase specified, for that file and before the execution of a CLOSE statement without the REEL or UNIT phrase for that file.

* **output procedure.** A set of statements to which control is given during execution of a SORT statement after the sort function is completed, or during execution of a MERGE statement after the merge function reaches a point at which it can select the next record in merged order when requested.

overflow condition. A condition that occurs when a portion of the result of an operation exceeds the capacity of the intended unit of storage.

P

packed decimal item. See "internal decimal item."

* **padding character.** An alphanumeric character used to fill the unused character positions in a physical record.

page. A vertical division of output data representing a physical separation of such data, the separation being based on internal logical requirements and/or external characteristics of the output medium.

* **page body.** That part of the logical page in which lines can be written and/or spaced.

* **paragraph.** In the Procedure Division, a paragraph-name followed by a separator period and by zero, one, or more sentences. In the IDENTIFICATION and ENVIRONMENT DIVISIONs, a paragraph header followed by zero, one, or more entries.

* **paragraph header.** A reserved word, followed by the separator period, that indicates the beginning of a paragraph in the IDENTIFICATION and ENVIRONMENT DIVISIONs. The permissible paragraph headers in the IDENTIFICATION DIVISION are:

```
PROGRAM-ID. (Program IDENTIFICATION DIVISION only)
CLASS-ID. (Class IDENTIFICATION DIVISION only)
METHOD-ID. (Method IDENTIFICATION DIVISION only)
AUTHOR.
INSTALLATION.
DATE-WRITTEN.
DATE-COMPILED.
SECURITY.
```

The permissible paragraph headers in the ENVIRONMENT DIVISION are:

```
SOURCE-COMPUTER.
OBJECT-COMPUTER.
SPECIAL-NAMES.
REPOSITORY. (Program or Class CONFIGURATION SECTION)
FILE-CONTROL.
I-O-CONTROL.
```

* **paragraph-name.** A user-defined word that identifies and begins a paragraph in the Procedure Division.

parameter. Parameters are used to pass data values between calling and called programs.

password. A unique string of characters that a program, computer operator, or user must supply to meet security requirements before gaining access to data.

* **phrase.** A phrase is an ordered set of one or more consecutive COBOL character-strings that form a portion of a COBOL procedural statement or of a COBOL clause.

* **physical record.** See "block."

pointer data item. A data item in which address values can be stored. Data items are explicitly defined as pointers with the USAGE IS POINTER clause. ADDRESS OF special registers are implicitly defined as pointer data items. Pointer data items can be compared for equality or moved to other pointer data items.

portability. The ability to transfer an application program from one application platform to another with relatively few changes to the source program.

preloaded. In COBOL this refers to COBOL programs that remain resident in storage under IMS instead of being loaded each time they are called.

* **prime record key.** A key whose contents uniquely identify a record within an indexed file.

* **priority-number.** A user-defined word which classifies sections in the Procedure Division for purposes of segmentation. Segment-numbers may contain only the characters '0','1', ... , '9'. A segment-number may be expressed either as a one- or two-digit number.

* **procedure.** A paragraph or group of logically successive paragraphs, or a section or group of logically successive sections, within the Procedure Division.

* **procedure branching statement.** A statement that causes the explicit transfer of control to a statement other than the next executable statement in the sequence in which the statements are written in the source program. The procedure branching statements are: ALTER, CALL, EXIT, EXIT PROGRAM, GO TO, MERGE, (with the OUTPUT PROCEDURE phrase), PERFORM and SORT (with the INPUT PROCEDURE or OUTPUT PROCEDURE phrase).

Procedure Division. One of the four main component parts of a COBOL program, class definition, or method definition. The Procedure Division contains instructions for solving a problem. The Program and Method Procedure Divisions may contain imperative statements, conditional statements, compiler directing statements, paragraphs, procedures, and sections. The Class Procedure Division contains only method definitions.

procedure integration. One of the functions of the COBOL optimizer is to simplify calls to performed procedures or contained programs.

PERFORM procedure integration is the process whereby a PERFORM statement is replaced by its

performed procedures. Contained program procedure integration is the process where a CALL to a contained program is replaced by the program code.

* **procedure-name.** A user-defined word that is used to name a paragraph or section in the Procedure Division. It consists of a paragraph-name (which may be qualified) or a section-name.

procedure-pointer data item. A data item in which a pointer to an entry point can be stored. A data item defined with the USAGE IS PROCEDURE-POINTER clause contains the address of a procedure entry point.

* **program identification entry.** An entry in the PROGRAM-ID paragraph of the IDENTIFICATION DIVISION which contains clauses that specify the program-name and assign selected program attributes to the program.

* **program-name.** In the IDENTIFICATION DIVISION and the end program header, a user-defined word that identifies a COBOL source program.

* **pseudo-text.** A sequence of text words, comment lines, or the separator space in a source program or COBOL library bounded by, but not including, pseudo-text delimiters.

* **pseudo-text delimiter.** Two contiguous equal sign characters (==) used to delimit pseudo-text.

* **punctuation character.** A character that belongs to the following set:

Character	Meaning
,	comma
;	semicolon
:	colon
.	period (full stop)
"	quotation mark
(left parenthesis
)	right parenthesis
	space
=	equal sign

Q

QSAM (Queued Sequential Access Method). An extended version of the basic sequential access method (BSAM). When this method is used, a queue is formed of input data blocks that are awaiting processing or of output data blocks that have been processed and are awaiting transfer to auxiliary storage or to an output device.

* **qualified data-name.** An identifier that is composed of a data-name followed by one or more sets of either of the connectives OF and IN followed by a data-name qualifier.

* **qualifier.**

1. A data-name or a name associated with a level indicator which is used in a reference either together with another data-name which is the name of an item that is subordinate to the qualifier or together with a condition-name.
2. A section-name that is used in a reference together with a paragraph-name specified in that section.
3. A library-name that is used in a reference together with a text-name associated with that library.

R

* **random access.** An access mode in which the program-specified value of a key data item identifies the logical record that is obtained from, deleted from, or placed into a relative or indexed file.

* **record.** See "logical record."

* **record area.** A storage area allocated for the purpose of processing the record described in a record description entry in the File Section of the DATA DIVISION. In the File Section, the current number of character positions in the record area is determined by the explicit or implicit RECORD clause.

* **record description.** See "record description entry."

* **record description entry.** The total set of data description entries associated with a particular record. The term is synonymous with record description.

recording mode. The format of the logical records in a file. Recording mode can be F (fixed-length), V (variable-length), S (spanned), or U (undefined).

record key. A key whose contents identify a record within an indexed file.

* **record-name.** A user-defined word that names a record described in a record description entry in the DATA DIVISION of a COBOL program.

* **record number.** The ordinal number of a record in the file whose organization is sequential.

recursion. A program calling itself or being directly or indirectly called by a one of its called programs.

recursively capable. A program is recursively capable (can be called recursively) if the RECURSIVE attribute is on the PROGRAM-ID statement.

reel. A discrete portion of a storage medium, the dimensions of which are determined by each implementor that contains part of a file, all of a file, or any number of files. The term is synonymous with unit and volume.

reentrant. The attribute of a program or routine that allows more than one user to share a single copy of a load module.

* **reference format.** A format that provides a standard method for describing COBOL source programs.

reference modification. A method of defining a new alphanumeric data item by specifying the leftmost character and length relative to the leftmost character of another alphanumeric data item.

* **reference-modifier.** A syntactically correct combination of character-strings and separators that defines a unique data item. It includes a delimiting left parenthesis separator, the leftmost character position, a colon separator, optionally a length, and a delimiting right parenthesis separator.

* **relation.** See "relational operator" or "relation condition."

* **relational operator.** A reserved word, a relation character, a group of consecutive reserved words, or a group of consecutive reserved words and relation characters used in the construction of a relation condition. The permissible operators and their meanings are:

Operator	Meaning
IS GREATER THAN	Greater than
IS >	Greater than
IS NOT GREATER THAN	Not greater than
IS NOT >	Not greater than
IS LESS THAN	Less than
IS <	Less than
IS NOT LESS THAN	Not less than
IS NOT <	Not less than
IS EQUAL TO	Equal to
IS =	Equal to
IS NOT EQUAL TO	Not equal to
IS NOT =	Not equal to
IS GREATER THAN OR EQUAL TO	Greater than or equal to
IS >=	Greater than or equal to
IS LESS THAN OR EQUAL TO	Less than or equal to
IS <=	Less than or equal to

* **relation character.** A character that belongs to the following set:

Character	Meaning
>	greater than
<	less than
=	equal to

* **relation condition.** The proposition, for which a truth value can be determined, that the value of an arithmetic expression, data item, nonnumeric literal, or index-name has a specific relationship to the value of another arithmetic expression, data item, nonnumeric literal, or index name. (See also "relational operator.")

* **relative file.** A file with relative organization.

* **relative key.** A key whose contents identify a logical record in a relative file.

* **relative organization.** The permanent logical file structure in which each record is uniquely identified by an integer value greater than zero, which specifies the record's logical ordinal position in the file.

* **relative record number.** The ordinal number of a record in a file whose organization is relative. This number is treated as a numeric literal which is an integer.

* **reserved word.** A COBOL word specified in the list of words that may be used in a COBOL source program, but that must not appear in the program as user-defined words or system-names.

* **resource.** A facility or service, controlled by the operating system, that can be used by an executing program.

* **resultant identifier.** A user-defined data item that is to contain the result of an arithmetic operation.

reusable environment. A reusable environment is when you establish an assembler program as the main program by using either ILBOSTP0 programs, IGZERRE programs, or the RTEREUS run-time option.

routine. A set of statements in a COBOL program that causes the computer to perform an operation or series of related operations. In &cel., refers to either a procedure, function, or subroutine.

* **routine-name.** A user-defined word that identifies a procedure written in a language other than COBOL.

* **run time.** The time at which an object program is executed. The term is synonymous with object time.

run-time environment. The environment in which a COBOL program executes.

* **run unit.** A stand-alone object program, or several object programs, that interact via COBOL CALL statements, which function at run time as an entity.

S

SBCS (Single Byte Character Set). See "Single Byte Character Set (SBCS)".

scope terminator. A COBOL reserved word that marks the end of certain Procedure Division statements. It may be either explicit (END-ADD, for example) or implicit (separator period).

* **section.** A set of zero, one or more paragraphs or entities, called a section body, the first of which is preceded by a section header. Each section consists of the section header and the related section body.

* **section header.** A combination of words followed by a separator period that indicates the beginning of a section in the Environment, Data, and Procedure Divisions. In the ENVIRONMENT and DATA DIVISIONs, a section header is composed of reserved words followed by a separator period. The permissible section headers in the ENVIRONMENT DIVISION are:

```
CONFIGURATION SECTION.  
INPUT-OUTPUT SECTION.
```

The permissible section headers in the DATA DIVISION are:

```
FILE SECTION.  
WORKING-STORAGE SECTION.  
LOCAL-STORAGE SECTION.  
LINKAGE SECTION.
```

In the Procedure Division, a section header is composed of a section-name, followed by the reserved word SECTION, followed by a separator period.

* **section-name.** A user-defined word that names a section in the Procedure Division.

selection structure. A program processing logic in which one or another series of statements is executed, depending on whether a condition is true or false.

* **sentence.** A sequence of one or more statements, the last of which is terminated by a separator period.

* **separately compiled program.** A program which, together with its contained programs, is compiled separately from all other programs.

* **separator.** A character or two contiguous characters used to delimit character-strings.

* **separator comma.** A comma (,) followed by a space used to delimit character-strings.

* **separator period.** A period (.) followed by a space used to delimit character-strings.

* **separator semicolon.** A semicolon (;) followed by a space used to delimit character-strings.

sequence structure. A program processing logic in which a series of statements is executed in sequential order.

* **sequential access.** An access mode in which logical records are obtained from or placed into a file in a consecutive predecessor-to-successor logical record sequence determined by the order of records in the file.

* **sequential file.** A file with sequential organization.

* **sequential organization.** The permanent logical file structure in which a record is identified by a predecessor-successor relationship established when the record is placed into the file.

serial search. A search in which the members of a set are consecutively examined, beginning with the first member and ending with the last.

* **77-level-description-entry.** A data description entry that describes a noncontiguous data item with the level-number 77.

* **sign condition.** The proposition, for which a truth value can be determined, that the algebraic value of a data item or an arithmetic expression is either less than, greater than, or equal to zero.

* **simple condition.** Any single condition chosen from the set:

- Relation condition
- Class condition
- Condition-name condition
- Switch-status condition
- Sign condition

Single Byte Character Set (SBCS). A set of characters in which each character is represented by a single byte. See also "EBCDIC (Extended Binary-Coded Decimal Interchange Code)."

slack bytes. Bytes inserted between data items or records to ensure correct alignment of some numeric items. Slack bytes contain no meaningful data. In some cases, they are inserted by the compiler; in others, it is the responsibility of the programmer to insert them. The SYNCHRONIZED clause instructs the compiler to insert slack bytes when they are needed for proper alignment. Slack bytes between records are inserted by the programmer.

SOM. *System Object Model*

* **sort file.** A collection of records to be sorted by a SORT statement. The sort file is created and can be used by the sort function only.

* **sort-merge file description entry.** An entry in the File Section of the DATA DIVISION that is composed of the level indicator SD, followed by a file-name, and then followed by a set of file clauses as required.

* **SOURCE-COMPUTER.** The name of an ENVIRONMENT DIVISION paragraph in which the computer environment, within which the source program is compiled, is described.

* **source computer entry.** An entry in the SOURCE-COMPUTER paragraph of the ENVIRONMENT DIVISION which contains clauses that describe the computer environment in which the source program is to be compiled.

* **source item.** An identifier designated by a SOURCE clause that provides the value of a printable item.

source program. Although it is recognized that a source program may be represented by other forms and symbols, in this document it always refers to a syntactically correct set of COBOL statements. A COBOL source program commences with the IDENTIFICATION DIVISION or a COPY statement. A COBOL source program is terminated by the end program header, if specified, or by the absence of additional source program lines.

* **special character.** A character that belongs to the following set:

Character	Meaning
+	plus sign
-	minus sign (hyphen)
*	asterisk
/	slant (virgule, slash)
=	equal sign
\$	currency sign
,	comma (decimal point)
;	semicolon
.	period (decimal point, full stop)
"	quotation mark
(left parenthesis
)	right parenthesis
>	greater than symbol
<	less than symbol
:	colon

* **special-character word.** A reserved word that is an arithmetic operator or a relation character.

SPECIAL-NAMES. The name of an ENVIRONMENT DIVISION paragraph in which environment-names are related to user-specified mnemonic-names.

* **special names entry.** An entry in the SPECIAL-NAMES paragraph of the ENVIRONMENT DIVISION which provides means for specifying the currency sign; choosing the decimal point; specifying symbolic characters; relating implementor-names to user-specified mnemonic-names; relating alphabet-names to character sets or collating

sequences; and relating class-names to sets of characters.

* **special registers.** Certain compiler generated storage areas whose primary use is to store information produced in conjunction with the use of a specific COBOL feature.

* **standard data format.** The concept used in describing the characteristics of data in a COBOL DATA DIVISION under which the characteristics or properties of the data are expressed in a form oriented to the appearance of the data on a printed page of infinite length and breadth, rather than a form oriented to the manner in which the data is stored internally in the computer, or on a particular external medium.

* **statement.** A syntactically valid combination of words, literals, and separators, beginning with a verb, written in a COBOL source program.

structured programming. A technique for organizing and coding a computer program in which the program comprises a hierarchy of segments, each segment having a single entry point and a single exit point. Control is passed downward through the structure without unconditional branches to higher levels of the hierarchy.

* **sub-class.** A class that inherits from another class. When two classes in an inheritance relationship are considered together, the sub-class is the inheritor or inheriting class; the *super-class* is the inheritee or inherited class.

* **subject of entry.** An operand or reserved word that appears immediately following the level indicator or the level-number in a DATA DIVISION entry.

* **subprogram.** See "called program."

* **subscript.** An occurrence number represented by either an integer, a data-name optionally followed by an integer with the operator + or -, or an index-name optionally followed by an integer with the operator + or -, that identifies a particular element in a table. A subscript may be the word ALL when the subscripted identifier is used as a function argument for a function allowing a variable number of arguments.

* **subscripted data-name.** An identifier that is composed of a data-name followed by one or more subscripts enclosed in parentheses.

* **super-class.** A class that is inherited by another class. See also *sub-class*.

switch-status condition. The proposition, for which a truth value can be determined, that an UPSI switch, capable of being set to an 'on' or 'off' status, has been set to a specific status.

* **symbolic-character.** A user-defined word that specifies a user-defined figurative constant.

syntax. (1) The relationship among characters or groups of characters, independent of their meanings or the manner of their interpretation and use. (2) The structure of expressions in a language. (3) The rules governing the structure of a language. (4) The relationship among symbols. (5) The rules for the construction of a statement.

* **system-name.** A COBOL word that is used to communicate with the operating environment.

System Object Model (SOM). IBM's object-oriented programming technology for building, packaging, and manipulating class libraries. SOM conforms to the Object Management Group's (OMG) Common Object Request Broker Architecture (CORBA) standards.

T

* **table.** A set of logically consecutive items of data that are defined in the DATA DIVISION by means of the OCCURS clause.

* **table element.** A data item that belongs to the set of repeated items comprising a table.

text deck. Synonym for *object deck* or *object module*.

* **text-name.** A user-defined word that identifies library text.

* **text word.** A character or a sequence of contiguous characters between margin A and margin R in a COBOL library, source program, or in pseudo-text which is:

- A separator, except for: space; a pseudo-text delimiter; and the opening and closing delimiters for nonnumeric literals. The right parenthesis and left parenthesis characters, regardless of context within the library, source program, or pseudo-text, are always considered text words.
- A literal including, in the case of nonnumeric literals, the opening quotation mark and the closing quotation mark that bound the literal.
- Any other sequence of contiguous COBOL characters except comment lines and the word 'COPY' bounded by separators that are neither a separator nor a literal.

top-down design. The design of a computer program using a hierarchic structure in which related functions are performed at each level of the structure.

top-down development. See "structured programming."

trailer-label. (1) A file or data set label that follows the data records on a unit of recording medium. (2) Synonym for end-of-file label.

* **truth value.** The representation of the result of the evaluation of a condition in terms of one of two values: true or false.

U

* **unary operator.** A plus (+) or a minus (-) sign, that precedes a variable or a left parenthesis in an arithmetic expression and that has the effect of multiplying the expression by +1 or -1, respectively.

unit. A module of direct access, the dimensions of which are determined by IBM.

universal object reference. A data-name that can refer to an object of any class.

* **unsuccessful execution.** The attempted execution of a statement that does not result in the execution of all the operations specified by that statement. The unsuccessful execution of a statement does not affect any data referenced by that statement, but may affect status indicators.

UPSI switch. A program switch that performs the functions of a hardware switch. Eight are provided: UPSI-0 through UPSI-7.

* **user-defined word.** A COBOL word that must be supplied by the user to satisfy the format of a clause or statement.

V

* **variable.** A data item whose value may be changed by execution of the object program. A variable used in an arithmetic expression must be a numeric elementary item.

* **variable length record.** A record associated with a file whose file description or sort-merge description entry permits records to contain a varying number of character positions.

* **variable occurrence data item.** A variable occurrence data item is a table element which is repeated a variable number of times. Such an item must contain an OCCURS DEPENDING ON clause in its data description entry, or be subordinate to such an item.

* **variably located group.** A group item following, and not subordinate to, a variable-length table in the same level-01 record.

* **variably located item.** A data item following, and not subordinate to, a variable-length table in the same level-01 record.

* **verb.** A word that expresses an action to be taken by a COBOL compiler or object program.

VM/SP (Virtual Machine/System Product). An IBM-licensed program that manages the resources of a single computer so that multiple computing systems appear to exist. Each virtual machine is the functional equivalent of a "real" machine.

volume. A module of external storage. For tape devices it is a reel; for direct-access devices it is a unit.

volume switch procedures. System specific procedures executed automatically when the end of a unit or reel has been reached before end-of-file has been reached.

W

* **word.** A character-string of not more than 30 characters which forms a user-defined word, a system-name, a reserved word, or a function-name.

* **WORKING-STORAGE SECTION.** The section of the DATA DIVISION that describes working storage data items, composed either of noncontiguous items or working storage records or of both.

Z

zoned decimal item. See "external decimal item."

List of Abbreviations

AIX	Advanced Interactive Executive from IBM	LAN	local area network
API	application program interface	LAPS	LAN adapter and protocol support
BMS	basic mapping support	MB	megabytes
CD	compact disc	MPTS	multiprotocol transport services
CICS	Customer Information Control System	MVS	multiple virtual storage
DB2	Data Base 2 from IBM	PDS	partitioned data set
DDCS	distributed database connection services	PPT	program properties table
DHCP	Dynamic Host Configuration Protocol	PTF	program temporary fix
DLL	dynamic link library	RCT	resource control table
DPL	distributed program link	RDBM	relational database manager
DRDA	distributed relational database architecture	ROM	read-only memory
ECI	external call interface	RPC	remote procedure call
EPI	external presentation interface	SDK	Software Developer's Kit
ETI	external transaction initiation	SIT	system initialization table
FAT	file allocation table	SMIT	System Management Interface Tool
GUI	graphical user interface	SOM	start-of-message code
HPFS	high-performance file system	SQL	Structured Query Language
IBM	International Business Machines Corporation	TCP/IP	Transmission Control Protocol/Internet Protocol
ISPF	interactive system productivity facility	TMP	terminal monitor program
ITSO	International Technical Support Organization	TSO	time-sharing option
		UPM	user profile management
		VDE	visual development environment

Index

A

- abbreviations 263
- acronyms 263
- AIX
 - CICS for AIX and OS/2 overview 131
 - environment overview 109

B

- bibliography 237

C

- CICS
 - CICS for AIX and OS/2 overview 131
 - CICS for AIX configuration 142
 - CICS for AIX installation 137
 - CICS for AIX overview 132
 - CICS for AIX server overview 133
 - client operation 74
 - client overview 58, 69
 - compile and linking CICS for AIX 145
 - copybooks 77
 - DCE installation 133
 - debugging programs 106
 - dynamic link library 77
 - ECL call 60
 - EPL call 60
 - ETI call 60
 - libraries and objects 77
 - OS/2 CICS client to CICS AIX 150
 - OS/2 CICS installation 70
 - running applications 77
 - server configuration 66
 - server installation 60
 - server local name 62
 - server overview 58
 - server setup 74
 - SQLLIB 77
 - using Transaction Assistant 102
- COBOL Set for AIX
 - installing and configuring 132
- Communications Manager
 - installation and configuration 158
 - MVS server with APPC configuration 215

D

- DB2
 - alias creation 102
 - client access from OS/2 to AIX 119
 - client configuration for AIX environment 125
 - client NetBIOS node name 37
 - client setup 37

DB2 (continued)

- communication protocol 18
- connecting to a database 42
- database cataloging 42
- DB2 for AIX overview 110
- DB2 for AIX server configuration 116
- DB2 for AIX server installation 110
- DB2 for AIX TCP/IP configuration 118
- DB2COMM environment variable 18, 42
- local logon 57
- NetBIOS resources 21
- server configuration 18
- server installation 13
- server node name 18
- server overview 1
- Single-User installation 31
- Single-User overview 2
- Software Developer's Kit installation 35
- Software Developer's Kit overview 2

L

- LAN Requester
 - installation 25
 - network messaging 30
- LAN Server
 - defining users 8
 - installation 2
 - overview 1
 - server name 5

M

- MVS
 - host offload overview 157
 - MVS server and VTAM definitions 227
 - MVS server with OS/2 client overview 213

N

- NetBIOS
 - DB2 resources 21
 - parameter settings 23

S

- sample application
 - EMPLLU client installation 95
 - EMPLLU host-workstation build and run 181
 - EMPLLU host-workstation CICS and DB2 184
 - EMPLLU overview 84
 - EMPLLU server installation 84
 - EMPLLUH host offload setup 168
 - EMPLLUH workstation preparation 171
 - IWZZ5 application overview 78

sample application (*continued*)

- IWZZ5 CICS client setup 81
- IWZZ5 CICS server setup 79
- SALESDEP compiler settings 53
- SALESDEP on OS/2 and AIX 129
- SALESDEP project setup 47

T

TCP/IP

- configuration 116
- TCP/IP installation on OS/2 119

V

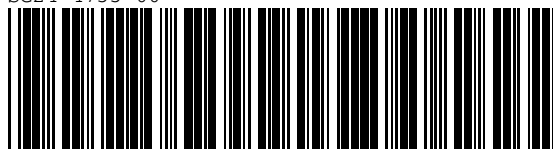
VisualAge COBOL

- installation 43
- overview 2



Printed in U.S.A.

SG24-4733-00



Artwork Definitions

<u>id</u>	<u>File</u>	<u>Page</u>	<u>References</u>
ITLOGO	4733SU		
ITLOGOS	4733SU	i	i
		i	

Figures

<u>id</u>	<u>File</u>	<u>Page</u>	<u>References</u>
CSODOVF	CSCBODOV	1	1
ODSV00	CSCBODSV	3	1, 1
ODSV01	CSCBODSV	4	3
ODSV02	CSCBODSV	4	3
ODSV03	CSCBODSV	5	4
ODSV04	CSCBODSV	5	4
ODSV05	CSCBODSV	5	5, 5
ODSV06	CSCBODSV	6	5
ODSV07	CSCBODSV	6	8
ODSV08	CSCBODSV	6	5, 9
ODSV09	CSCBODSV	7	9
ODSV10	CSCBODSV	7	10
ODSV11	CSCBODSV	8	6
ODSV12	CSCBODSV	8	11
ODSV13	CSCBODSV	9	6
ODSV14	CSCBODSV	9	12
ODSV15	CSCBODSV	9	7
ODSV16	CSCBODSV	10	7
ODSV17	CSCBODSV	10	13
ODSV18	CSCBODSV	10	8
ODSV19	CSCBODSV	11	8
ODSV20	CSCBODSV	11	15
ODSV21	CSCBODSV	12	8
ODSV22	CSCBODSV		16

		12	24	12, 13
ODSV23	CSCBODSV			
		13	25	13
ODSV24	CSCBODSV			
		14	26	13
ODSV25	CSCBODSV			
		14	27	14, 16
ODSV26	CSCBODSV			
		15	28	14
ODSV27	CSCBODSV			
		15	29	15
ODSV28	CSCBODSV			
		16	30	15
ODSV29	CSCBODSV			
		16	31	16
ODSV30	CSCBODSV			
		16	32	16
ODSV31	CSCBODSV			
		17	33	16, 17, 18
ODSV32	CSCBODSV			
		17	34	17
ODSV33	CSCBODSV			
		18	35	17
ODSV34	CSCBODSV			
		18	36	18
ODSV35	CSCBODSV			
		18	37	
ODSV36	CSCBODSV			
		19	38	19
ODSV37	CSCBODSV			
		19	39	19
ODSV38	CSCBODSV			
		20	40	19
ODSV39	CSCBODSV			
		20	41	20
ODSV40	CSCBODSV			
		21	42	20
CBNW23	CSCBODSV			
		21	43	21
CBNW24	CSCBODSV			
		22	44	21
CBNW25	CSCBODSV			
		22	45	22
CBNW27	CSCBODSV			
		23	46	22
CBNW28	CSCBODSV			
		24	47	23
CBNW29	CSCBODSV			
		24	48	24
CBNW30	CSCBODSV			
		25	49	24
ODCL00	CSCBODCL			
		26	50	26
ODCL01	CSCBODCL			
		27	51	26
ODCL02	CSCBODCL			
		27	52	27

ODCL03	CSCBODCL	28	53	27
ODCL04	CSCBODCL	28	54	28
ODCL05	CSCBODCL	29	55	28
ODCL39	CSCBODCL	30	56	29
ODCL06	CSCBODCL	31	57	30
ODCL07	CSCBODCL	31	58	31
ODCL08	CSCBODCL	32	59	32
ODCL09	CSCBODCL	32	60	32
ODCL10	CSCBODCL	33	61	33
ODCL11	CSCBODCL	33	62	33
ODCL12	CSCBODCL	34	63	33
ODCL13	CSCBODCL	34	64	34
ODCL14	CSCBODCL	34	65	34
ODCL15	CSCBODCL	35	66	34
ODCL70	CSCBODCL	35	67	35
ODCL71	CSCBODCL	36	68	36
ODCL72	CSCBODCL	36	69	36
ODCL73	CSCBODCL	37	70	36
ODCL74	CSCBODCL	37	71	37
ODCL30	CSCBODCL	38	72	38
ODCL31	CSCBODCL	38	73	38
ODCL32	CSCBODCL	39	74	39
ODCL33	CSCBODCL	39	75	39
ODCL34	CSCBODCL	40	76	40
ODCL35	CSCBODCL	41	77	40, 40
ODCL36	CSCBODCL	41	78	41
ODCL80	CSCBODCL	42	79	41
ODCL37	CSCBODCL	42	80	42
CBNW00	CSCBODCL			

		44	81	44
CBNW01	CSCBODCL			
		44	82	44
ODCL18	CSCBODCL			
		45	83	44
ODCL19	CSCBODCL			
		45	84	45
ODCL22	CSCBODCL			
		46	85	45
ODCL23	CSCBODCL			
		46	86	46
ODCL24	CSCBODCL			
		47	87	46
ODCL25	CSCBODCL			
		47	88	47
CBNW02	CSCBODCB			
		48	89	48
ODCL61	CSCBODCB			
		48	90	48
ODCL62	CSCBODCB			
		49	91	48
ODCL63	CSCBODCB			
		49	92	49
ODCL55	CSCBODCB			
		50	93	50
ODCL64	CSCBODCB			
		50	94	50
ODCL65	CSCBODCB			
		51	95	51
ODCL58	CSCBODCB			
		51	96	51
ODCL66	CSCBODCB			
		52	97	51
ODCL43	CSCBODCB			
		53	98	52, 53, 56
CBNW03	CSCBODCB			
		53	99	53
CBNW04	CSCBODCB			
		54	100	54
CBNW05	CSCBODCB			
		55	101	54
CBNW06	CSCBODCB			
		55	102	55, 93
CBNW07	CSCBODCB			
		56	103	55
ODCL67	CSCBODCB			
		57	104	56
ODCL47	CSCBODCB			
		57	105	57
ODCL48	CSCBODCB			
		58	106	57
ODCL49	CSCBODCB			
		58	107	58
CSOCO VF	CSCBOCOV			
		59	108	59
OCSV00	CSCBOCSV			
		61	109	

OCSV01	CSCBOCSV		60	
		61	110	
OCSV02	CSCBOCSV		61	
		62	111	
OCSV03	CSCBOCSV		61	
		62	112	
OCSV04	CSCBOCSV		62	
		63	113	
OCSV05	CSCBOCSV		62, 74	
		63	114	
OCSV06	CSCBOCSV		63	
		63	115	
OCSV08	CSCBOCSV		63	
		64	116	
OCSV07	CSCBOCSV		64	
		65	117	
OCSV09	CSCBOCSV		64, 64, 179	
		66	118	
OCSV11	CSCBOCSV		65	
		67	119	
OCSV12	CSCBOCSV		66	
		67	120	
CBNW15	CSCBOCSV		67	
		68	121	
CBNW14	CSCBOCSV		68	
		69	122	
OCCL80	CSCBOCCI		68	
		71	123	
OCCL81	CSCBOCCI		70	
		71	124	
OCCL82	CSCBOCCI		71	
		71	125	
OCCL83	CSCBOCCI		71	
		72	126	
OCCL84	CSCBOCCI		72	
		72	127	
OCCL85	CSCBOCCI		73	
		73	128	
OCCL86	CSCBOCCI		73	
		73	129	
OCCL87	CSCBOCCI		73	
		73	130	
OCCL11	CSCBOCCL		73	
		75	131	
OCCL12	CSCBOCCL		75	
		76	132	
OCCL13	CSCBOCCL		75	
		77	133	
OCCL20	CSCBOCCL		76	
		79	134	
CBNW08	CSCBOCCL		79	
		80	135	
CBNW09	CSCBOCCL		79	
		80	136	
OCCLB0	CSCBOCCL		80	
		81	137	
OCCLB1	CSCBOCCL		81	

		82	138	81
OCCLB2	CSCBOCCL			
		83	139	82
OCCLB3	CSCBOCCL			
		84	140	83
OCCL30	CSCBOCCL			
		85	141	85
OCCL31	CSCBOCCL			
		86	142	85
OCCL32	CSCBOCCL			
		86	143	86
OCCL33	CSCBOCCL			
		87	144	86
OCCL34	CSCBOCCL			
		87	145	87
OCCL52	CSCBOCCL			
		88	146	87
OCCL35	CSCBOCCL			
		88	147	88
OCCL36	CSCBOCCL			
		89	148	89
OCCL51	CSCBOCCL			
		89	149	89
OCCL37	CSCBOCCL			
		90	150	90
OCCL38	CSCBOCCL			
		91	151	90
OCCL39	CSCBOCCL			
		91	152	91, 91
OCCLA0	CSCBOCCL			
		92	153	91, 100
CBNW10	CSCBOCCL			
		92	154	92, 94
CBNW11	CSCBOCCL			
		93	155	92
CBNW12	CSCBOCCL			
		94	156	93
CBNW13	CSCBOCCL			
		94	157	94
OCCLA4	CSCBOCCL			
		96	158	95
OCCLA5	CSCBOCCL			
		96	159	96, 98
OCCLA6	CSCBOCCL			
		97	160	97
OCCL47	CSCBOCCL			
		98	161	97
OCCL48	CSCBOCCL			
		98	162	98
OCCL49	CSCBOCCL			
		99	163	98
OCCL50	CSCBOCCL			
		100	164	99
CBNW16	CSCBOCCL			
		101	165	101
OCCL88	CSCBOCCL			
		101	166	

				101
CBNW20	CSCBOCTA			
		103	167	103
CBNW22	CSCBOCTA			
		104	168	103
CBNW21	CSCBOCTA			
		104	169	104
ADOV	CSCBADSV			
		110	170	109
ADSV00	CSCBADSV			
		111	171	111
ADSV01	CSCBADSV			
		112	172	111
ADSV02	CSCBADSV			
		113	173	112
ADSV03	CSCBADSV			
		114	174	113
ADSV04	CSCBADSV			
		115	175	115
ADSV05	CSCBADSV			
		115	176	115
ADSV06	CSCBADSV			
		116	177	115, 129
ADSV07	CSCBADSV			
		117	178	116
ADSV08	CSCBADSV			
		118	179	117
ADCL15	CSCBADCL			
		119	180	119
ADCL00	CSCBADCL			
		120	181	119
ADCL01	CSCBADCL			
		120	182	120
ADCL02	CSCBADCL			
		121	183	120
ADCL03	CSCBADCL			
		121	184	121
ADCL04	CSCBADCL			
		122	185	121
ADCL05	CSCBADCL			
		122	186	122
ADCL17	CSCBADCL			
		123	187	123
ADCL07	CSCBADCL			
		123	188	
ADCL08	CSCBADCL			
		124	189	124
ADCL09	CSCBADCL			
		124	190	124
ADCL10	CSCBADCL			
		125	191	
ADCL18	CSCBADCL			
		125	192	125
ADCL11	CSCBADCL			
		126	193	126
ADCL12	CSCBADCL			
		127	194	126, 126
ADCL19	CSCBADCL			
		127	195	

ADCL13	CSCBADCL			127
		128	196	
ADCL20	CSCBADCL			127
		128	197	
ADCL14	CSCBADCL			128
		129	198	
ACOV	CSCBACOV			
		131	199	
ACSV05	CSCBACSV			131
		135	200	
ACSV06	CSCBACSV			135
		136	201	
ACSV07	CSCBACSV			135
		137	202	
ACSV17	CSCBACSV			136
		140	203	
ACSV09	CSCBACSV			139
		142	204	
ACSV16	CSCBACSV			141
		144	205	
ACSV10	CSCBACSV			144
		145	206	
ACSV15	CSCBACSV			143
		146	207	
ACSV11	CSCBACSV			145
		147	208	
ACSV12	CSCBACSV			147
		148	209	
ACSV13	CSCBACSV			147
		149	210	
ACCL04	CSCBACSV			149
		151	211	
ACCL00	CSCBACSV			151
		152	212	
ACCL01	CSCBACSV			151
		153	213	
ACCL02	CSCBACSV			152
		154	214	
ACCL03	CSCBACSV			153
		155	215	
CSMOOVF	CSCBMOOV			155
		158	216	
MOCL00	CSCBMOCM			157, 158
		159	217	
MOCL01	CSCBMOCM			159, 216
		160	218	
MOCL02	CSCBMOCM			159
		160	219	
MOCL03	CSCBMOCM			160, 216
		161	220	
MOCL04	CSCBMOCM			160
		161	221	
MOCL06	CSCBMOCM			161
		162	222	
MOCL07	CSCBMOCM			161
		162	223	
MOCL08	CSCBMOCM			162

		163	224	162
MOCL10	CSCBMOCM	163	225	163
MOCL11	CSCBMOCM	164	226	163
MOCL12	CSCBMOCM	164	227	164
MOCL13	CSCBMOCM	165	228	164
MOCL14	CSCBMOCM	165	229	165, 216
MOCL15	CSCBMOCM	166	230	165
MOCL16	CSCBMOCM	166	231	166, 167
MOCL17	CSCBMOCM	167	232	166
MOCL18	CSCBMOCM	167	233	167
MODL00	CSCBMODL	170	234	169
MODL01	CSCBMODL	170	235	170
MOWS00	CSCBMOWS	172	236	171
MOWS01	CSCBMOWS	172	237	172
MOWS02	CSCBMOWS	173	238	172
MOWS03	CSCBMOWS	173	239	173
MOWS04	CSCBMOWS	174	240	173
MOWS05	CSCBMOWS	174	241	174
MOWS06	CSCBMOWS	175	242	174
CBNW32	CSCBMOWS	176	243	175
CBNW33	CSCBMOWS	176	244	176
MOWS09	CSCBMOWS	178	245	177
MOWS10	CSCBMOWS	179	246	179
MOWS11	CSCBMOWS	180	247	179
MOWS12	CSCBMOWS	180	248	180
MOWS13	CSCBMOWS	182	249	181
MOWS14	CSCBMOWS	183	250	182
MOWS15	CSCBMOWS	184	251	183
MOWS16	CSCBMOWS	186	252	

				186
MOWS17	CSCBMOWS			
		186	253	186
MOWS18	CSCBMOWS			
MOWS19	CSCBMOWS	187	254	
		187	255	187
MOWS20	CSCBMOWS			
		188	256	187
MOWS21	CSCBMOWS			
		188	257	188
MOCL19	CSCBMOUL			
		190	258	190
MOCL20	CSCBMOUL			
		191	259	190
MOCL25	CSCBMOUL			
		191	260	191
MOCL21	CSCBMOUL			
		192	261	191
MOCL22	CSCBMOUL			
		192	262	192
MOCL23	CSCBMOUL			
		193	263	192
MOHS00	CSCBMOHS			
		196	264	195
MOHS01	CSCBMOHS			
		197	265	196, 209
MOHS02	CSCBMOHS			
		198	266	197
MOHS10	CSCBMOHS			
		199	267	
MOHS11	CSCBMOHS			
		200	268	
MOHS12	CSCBMOHS			
		202	269	
MOHS13	CSCBMOHS			
		203	270	
MOHS20	CSCBMOHS			
		205	271	205
MOHS21	CSCBMOHS			
		206	272	205, 206
MOHS22	CSCBMOHS			
		207	273	206
MOHS23	CSCBMOHS			
		208	274	207, 232
MOHS24	CSCBMOHS			
		209	275	209
MOHS25	CSCBMOHS			
		210	276	210
MOHS26	CSCBMOHS			
		211	277	210
CSMCOVF	CSCBMCOV			
		214	278	213
MCCM00	CSCBMCCM			
		216	279	216, 225
MCCM01	CSCBMCCM			
		217	280	216
MCCM02	CSCBMCCM			
		217	281	217
MCCM03	CSCBMCCM			
		218	282	

MCCM04	CSCBMCCM	218	283	217, 219, 219, 222
MCCM05	CSCBMCCM	219	284	218, 229
MCCM06	CSCBMCCM	220	285	219, 228, 229
MCCM07	CSCBMCCM	220	286	219
MCCM08	CSCBMCCM	221	287	220
MCCM09	CSCBMCCM	222	288	220, 229
MCCM15	CSCBMCCM	222	289	221
MCCM10	CSCBMCCM	223	290	222
MCCM11	CSCBMCCM	223	291	222, 223
MCCM12	CSCBMCCM	224	292	223, 225
MCCM13	CSCBMCCM	224	293	223, 225
MCCM14	CSCBMCCM	225	294	224
MCCC00	CSCBMCCC	226	295	225
MCVT00	CSCBMCVT	228	296	226
MCVT01	CSCBMCVT	228	297	227
MCVT02	CSCBMCVT	229	298	228
MCVT03	CSCBMCVT	229	299	228
MCCE00	CSCBMCCE	231	300	229
MCCE01	CSCBMCCE	232	301	230
MCCE02	CSCBMCCE	232	302	231
MCRA00	CSCBMCRA	234	303	232
				233

Headings			
----------	--	--	--

<u>id</u>	<u>File</u>	<u>Page</u>	<u>References</u>
CSOD	CSCBODOV	1	Chapter 1, OS/2 Local Area Network with DB2 59, 59, 109, 129, 129, 139
CSODSV	CSCBODSV	2	1.1, OS/2 Server Installation
ODSV1	CSCBODSV	3	1.1.1, Installing OS/2 LAN Server 4.0
ODSV2	CSCBODSV	8	1.1.2, Defining the Users of the Domain
ODSV3	CSCBODSV	13	1.1.3, Installing IBM DATABASE 2 for OS/2 Server
ODSV4	CSCBODSV	18	1.1.4, Configuring DB2 for OS/2 on the Server 37, 38

CSODCL	CSCBODCL	25	1.2, OS/2 Client Installation
ODCL2	CSCBODCL	31	1.2.1, Installing DB2 for OS/2 - Single-User
ODCL4	CSCBODCL	35	1.2.2, Installation of Software Developer's Kit for OS/2
ODCL3	CSCBODCL	37	1.2.3, DB2 Client Setup 19, 95
ODCL5	CSCBODCL	44	1.2.4, Installing IBM VisualAge for COBOL for OS/2
EXCB	CSCBODCB	47	1.2.5, Executing a COBOL Application That Accesses a Remote Database
CSOC	CSCBOCOV	59	Chapter 2, OS/2 LAN with CICS and DB2 132, 158, 213, 214
OCSV	CSCBOCSV	60	2.1, CICS for OS/2
OCSVCI	CSCBOCSV	60	2.1.1, CICS for OS/2 Installation
CSOCCL	CSCBOCCL	69	2.2, OS/2 CICS Client 103
OCCLIN	CSCBOCCI	70	2.2.1, CICS Client Installation 226
OCCL2	CSCBOCCL	74	2.2.2, Using OS/2 CICS Client
CSOCCS	CSCBOCCL	77	2.3, Running CICS Client/Server Applications
OCCL3	CSCBOCCL	78	2.3.1, Running the First CICS Client/Server Application
OCCL4	CSCBOCCL	79	2.3.1.1, Handling the Application on the CICS Server
OCCL5	CSCBOCCL	81	2.3.1.2, Handling the Application on the CICS Client
OCCL6	CSCBOCCL	84	2.3.2, Running a CICS Client/Server Application with Database Access 168, 181, 184
OCCL7	CSCBOCCL	85	2.3.2.1, Installing the Server Part of the Application 171, 185
OCCL8	CSCBOCCL	95	2.3.2.2, Installing the Client Part of the Application 181, 213, 233
OCCL9	CSCBOCCL	100	2.3.2.3, Running the Application 182, 233
OCCL10	CSCBOCTA	102	2.3.3, How to Use the Transaction Assistant 227
CSOCDE	CSCBOCDE	106	2.3.4, Debugging IBM VisualAge for COBOL for OS/2 CICS Programs 184
ADOV	CSCBADSV	109	Chapter 3, COBOL with DB2 for AIX and DB2 Client 143, 145
ADDBA	CSCBADSV	110	3.1, DB2 for AIX
ADSVIN	CSCBADSV	111	3.1.1, DB2 for AIX Server Installation
ADSVCF	CSCBADSV	116	3.1.2, DB2 for AIX Server Configuration
ADSVTC	CSCBADSV	116	3.1.2.1, TCP/IP Configuration
ADSVDC	CSCBADSV	118	3.1.2.2, DB2 for AIX Configuration for TCP/IP
ADCL	CSCBADCL	119	3.2, Set up DB2 Client to Access DB2 for AIX 119
ADCLTCP	CSCBADCL	119	3.2.1, TCP/IP Installation
ADCLDBC	CSCBADCL	125	3.2.2, DB2 Client Configuration
ADEX	CSCBADCL	129	3.3, Executing a COBOL Program That Accesses Database on AIX
AC	CSCBACOV	131	Chapter 4, COBOL with CICS for AIX and CICS Client for OS/2
ACCA	CSCBACSV	132	4.1, CICS for AIX Server Setup

ACCBIN	CSCBACSV	132	4.1.1, Installing and Configuring IBM COBOL Set for AIX
ACICDCE	CSCBACSV	133	4.1.2, Installing and Configuring CICS for AIX Server
ACDCEIN	CSCBACSV	133	4.1.2.1, Installing DCE
ACOSSET	CSCBACSV	134	4.1.2.2, Preparing Operating System for CICS
ACCAIN	CSCBACSV	137	4.1.2.3, Installing CICS for AIX
ACCACF	CSCBACSV	138	4.1.2.4, Configuring DCE
ACDBCF	CSCBACSV	138	4.1.2.5, Configuring DB2 for Queue and File Management
ACRGCS	CSCBACSV	140	4.1.2.6, Setting up CICS Region
ACCSCF	CSCBACSV	143	4.1.2.7, Configuring CICS for AIX Server
ACCLCS	CSCBACSV	145	4.1.3, Compiling and Linking the CICS Server Program
ACDCCP	CSCBACSV	147	4.1.4, Defining CICS Resources for the CICS Server Program
ACCCS	CSCBACSV	150	4.2, CICS Client for OS/2 Environment Setup
ACCFCC	CSCBACSV	150	4.2.1, Configuring CICS Client to Connect to CICS for AIX
ACCLCC	CSCBACSV	154	4.2.2, Compiling and Linking the CICS Client Program
ACRCCS	CSCBACSV	155	4.2.3, Executing the CICS Client/Server Programs
CSMO	CSCBMOOV	157	Chapter 5, Host Development Offload 213, 214
CSMO1	CSCBMOCM	158	5.1, Installing and Configuring Communications Manager/2 for OS/2 Warp 214, 216
CSMO2	CSCBMODL	168	5.2, Download the Necessary Data to the Workstation 157
CSMO3	CSCBMODL	169	5.2.1, Source Code
CSMO4	CSCBMODL	171	5.2.2, Test Data
CSMO5	CSCBMOWS	171	5.3, Prepare the Application on the Workstation
CSMO6	CSCBMOWS	171	5.3.1, CICS 3270 Client Application
CSMO61	CSCBMOWS	171	5.3.1.1, Create the Project for the Client on the Workstation
CSMO62	CSCBMOWS	176	5.3.1.2, Adapt the Source Code of the Client Application
CSMO63	CSCBMOWS	177	5.3.1.3, Create the BMS Maps for CICS for OS/2
CSMO64	CSCBMOWS	178	5.3.1.4, CICS for OS/2 Definitions
CSMO65	CSCBMOWS	181	5.3.1.5, Build and Run the CICS Client
CSMO7	CSCBMOWS	184	5.3.2, CICS-DB2 Server Application 213
CSMO71	CSCBMOWS	185	5.3.2.1, Create the Project for the Server on the Workstation
CSMO72	CSCBMOWS	186	5.3.2.2, Create the Copybook for the DB2 Table
CSMO73	CSCBMOWS	188	5.3.2.3, Adapt the Source Code of the Server Application
CSMO74	CSCBMOWS	189	5.3.2.4, Build the Server Application
CSMO8	CSCBMOUL	189	5.4, Upload the Necessary Data to the MVS System
CSMO9	CSCBMOHS	193	5.5, Prepare the Application on the MVS System 233
CSMO10	CSCBMOHS	194	5.5.1, Adapt the Source Code for MVS
CSMO101	CSCBMOHS	194	5.5.1.1, Change the Map Definition
CSMO102	CSCBMOHS	194	5.5.1.2, Change the Communication Area Definition
CSMO103	CSCBMOHS	195	5.5.1.3, Change the Server Program Source
CSMO104	CSCBMOHS	195	5.5.1.4, Create the Copybook containing the Table Definition
CSMO11	CSCBMOHS		

CSMO111	CSCBMOHS	198	5.5.2, Build the MVS Application
CSMO112	CSCBMOHS	199	5.5.2.1, Create the BMS Maps for CICS for MVS/ESA
CSMO113	CSCBMOHS	200	5.5.2.2, Compile and Link the Client Program
CSMO114	CSCBMOHS	201	5.5.2.3, Compile the Service Calculation Routine
CSMO12	CSCBMOHS	202	5.5.2.4, Compile, Link, and Bind the Server Program
CSMO13	CSCBMOHS	204	5.5.3, CICS for MVS/ESA Definitions
CSMC	CSCBMCOV	208	5.5.4, Run the Transaction on the Host
CSMC1	CSCBMCOV	213	Chapter 6, OS/2 Client with MVS Server
CSMC2	CSCBMCCM	214	6.1, Overview
CSMC3	CSCBMCCM	215	6.2, Configuring Communications Manager/2 for OS/2 Warp
CSMC4	CSCBMCCM	216	6.2.1, APPC Set-up
CSMC5	CSCBMCCC	225	6.2.2, Change Number of Sessions
CSMC6	CSCBMCVT	226	6.3, Change the CICS Client Definitions
CSMC7	CSCBMCVT	227	6.4, VTAM Definitions
CSMC8	CSCBMCVT	227	6.4.1, NETID Definition
CSMC9	CSCBMCVT	227	6.4.2, XID/PU/LU Definition
		228	6.4.3, APPL Definition
CSMC10	CSCBMCVT	228	6.4.4, LOGMODE Definition
CSMC11	CSCBMCVT	229	6.4.5, VTAM Cross-Domain Environment
CSMC12	CSCBMCCE	229	6.5, CICS for MVS/ESA Definitions
CSMC13	CSCBMCCE	230	6.5.1, SIT Definitions
CSMC14	CSCBMCCE	230	6.5.2, Connection/Session Definitions
CSMC15	CSCBMCCE	232	6.5.3, Data Conversion Definitions
CSMC16	CSCBMCRA	233	6.6, Run the Application
NOTICES	SG244733 SCRIPT	235	Appendix A, Special Notices
			ii
BIBL	4733BIBL	237	Appendix B, Related Publications
ORDER	REDB\$ORD	239	How To Get ITSO Redbooks
			237

Tables

<u>id</u>	<u>File</u>	<u>Page</u>	<u>References</u>
MCOV00	CSCBMCOV	215	1
			215

Processing Options

Runtime values:

Document fileid	SG244733 SCRIPT
Document type	USERDOC
Document style	REDBOOK
Profile	EDFPRF40
Service Level	0022
SCRIPT/VS Release	4.0.0
Date	96.12.20
Time	18:02:13
Device	3820A
Number of Passes	3
Index	YES
SYSVAR D	YES
SYSVAR G	INLINE
SYSVAR X	YES

Formatting values used:

Annotation	NO
Cross reference listing	YES
Cross reference head prefix only	NO
Dialog	LABEL
Duplex	YES
DVCF conditions file	(none)
DVCF value 1	(none)
DVCF value 2	(none)
DVCF value 3	(none)
DVCF value 4	(none)
DVCF value 5	(none)
DVCF value 6	(none)
DVCF value 7	(none)
DVCF value 8	(none)
DVCF value 9	(none)
Explode	NO
Figure list on new page	YES
Figure/table number separation	YES
Folio-by-chapter	NO
Head 0 body text	Part
Head 1 body text	Chapter
Head 1 appendix text	Appendix
Hyphenation	NO
Justification	NO
Language	ENGL
Keyboard	395
Layout	OFF
Leader dots	YES
Master index	(none)
Partial TOC (maximum level)	4
Partial TOC (new page after)	INLINE
Print example id's	NO
Print cross reference page numbers	YES
Process value	(none)
Punctuation move characters	,
Read cross-reference file	(none)
Running heading/footer rule	NONE
Show index entries	NO
Table of Contents (maximum level)	3
Table list on new page	YES
Title page (draft) alignment	RIGHT
Write cross-reference file	(none)

Imbed Trace

Page 0	4733SU
Page 0	4733VARS
Page 0	REDB\$SJC
Page i	REDB\$ED1
Page i	4733EDNO
Page i	REDB\$ED2
Page xi	4733ABST
Page xi	4733ORG
Page xi	4733ACKS
Page xii	REDB\$COM
Page xii	4733MAIN
Page xii	CSCBODOV
Page 2	CSCBODSV
Page 25	CSCBODCL
Page 47	CSCBODCB
Page 58	CSCBOCOV
Page 60	CSCBOCSV
Page 69	CSCBOCCL
Page 70	CSCBOCCI
Page 102	CSCBOCTA
Page 106	CSCBOCDE
Page 107	CSCBADSV
Page 119	CSCBADCL
Page 130	CSCBACOV
Page 132	CSCBACSV
Page 155	CSCBMOOV
Page 158	CSCBMOCM
Page 167	CSCBMODL
Page 171	CSCBMOWS
Page 189	CSCBMOUL
Page 193	CSCBMOHS
Page 211	CSCBMCOV
Page 215	CSCBMCCM
Page 225	CSCBMCCC
Page 227	CSCBMCVT
Page 229	CSCBMCCE
Page 233	CSCBMCRA
Page 235	4733SPEC
Page 235	REDB\$SPE
Page 235	4733TMKS
Page 236	4733BIBL
Page 237	REDB\$BIB
Page 238	REDB\$ORD
Page 241	IGY99GLO
Page 261	4733ABRV